Variational Models and Algorithms for Blind Image Deconvolution with Applications

by

Bryan M. Williams

under the supervision of

Professor Ke Chen

UNIVERSITY OF LIVERPOOL

Thesis submitted in accordance with the requirements
of the University of Liverpool for the
degree of Doctor in Philosophy.

October 2014

Contents

Acknowledgement

I should like to express my gratitude to all of those people who helped and supported me in the completion of this thesis.

I should like to express my gratitude to my supervisor Prof. Ke Chen for his guidance and support throughout my doctoral studies. I should also like to thank other members of the mathematical sciences department for their advice and constructive criticism of my work: Prof. Natalia Movchan, Dr. Gayanne Piliposyan, Prof. Bakhti Vasiev and Dr. Rachel Bearon as well as Dr. Yalin Zheng and Prof. Simon Harding at St Paul's Eye unit and Prof. Dr. Joachim Weickert at Universität des Saarlandes.

I should like to thank my colleagues Dr. Lavdie Rada, Mazlinda Ibrahim, Dr. Behzad Ghanbari, Jack Spencer, Gemma Cook, Dr. Yalin Zheng, Prof. Xia Zhou, Dr. Li Sun, Dr. Faisal Fairag, Dr. Fabianna Zama and Dr. Paul Harris for all of the very interesting discussions we have had during this time as well as Madina Boshtayeva, Christopher Schroers, Sebastian Hoffmann, Nico Persch and Yan Zhang.

My most sincere appreciation to my family Nicole, Tyrone and James for their lifelong support and encouragement and to my great friends Dr. Lavdie Rada, Dr. Ian M. Bradford, Mazlinda Ibrahim, Jeffrey Brown and John Hewitt, whose friendship and support made my life as graduate student much easier to handle and contributed immeasurably to my enjoyment of this experience.

Last but by no means least, I recognise that this research would not have been possible without the academic and technical support of the Department of Mathematical Sciences, University of Liverpool and its staff. Also, thanks to the EPSRC for the financial support provided for my doctoral studies and to the Mathematical Image Analysis Group at Universität des Saarlandes.

Abstract

This thesis deals with numerical solutions to partial differential equations (PDEs) and their application in image processing, particularly image deblurring. The PDEs we deal with arise from the minimisation of variational models for techniques for image restoration from a single image (such as denoising [29, 56, 169, 180], deblurring [149, 9, 44, 94, 119, 178, 54, 176, 212] or inpainting [6, 16, 25, 52]) or reconstruction from several images (such as focus fusion [80, 128, 159] and many techniques aimed at super-resolution [107, 130, 146]) as well as the identification of objects (such as global and selective segmentation [53, 89, 162, 163]) and other tasks such as the restoration of an image from a limited selection of points to facilitate image compression [101, 158, 175]. The aim of image denoising is to remove noise corruption from an image and restore the true image, while the aim of image segmentation is to distinguish the foreground from the background of an image or to select a particular feature in an image automatically. Image deblurring, or deconvolution, aims to restore an image which has been corrupted by blur and noise which remains a particular problem in many areas including remote sensing, medical imaging, and consumer photography.

Deblurring tasks can be categorised into 3 types, all of which remain challenging subjects. *Non-blind deblurring* [15, 18, 152, 207] assumes that we know and can model the cause or degradation of the image precisely. The aim is then to recover the hidden true image using the knowledge of the blur function which is not a trivial task, particularly in the presence of strong blur, limited boundary information and noise. In contrast, *blind image deconvolution (BID)* [54, 31, 117, 61, 209] is the technique of recovering an image from blur degradation with no assumptions about the blur function. Such techniques commonly involve attempting to minimise a BID functional in order to recover both the true image and blur function simultaneously. Others involve attempting to use multiple images or image statistics in order to gain some knowledge of the blur function before deblurring the image. *Semi-blind deblurring* [123, 11, 2, 211] involves recovering an image from blur degradation making some assumptions about the blur function. For example, we might be able to make the assumption from observation that an image has been corrupted by motion blur. The task then would be to estimate the orientation and strength of the blur while recovering the image. Such techniques are often regarded as blind since crucial information is still not known. For example, the above problem might be called *blind motion deblurring*.

While there has been much research in the restoration of images, the performance of such methods remains poor particularly when the level of noise or blur is high. Many techniques also suffer from slow implementation. Identification, whether automatic or visual, of the blurring function can also prove a challenging task. While it is sometimes but not always possible to identify the type of blur function (for example Gaussian or motion blur) there still remains the challenge of identifying the level or amount of blur. It is also often the case that several types of blur are present and that the image cannot be recovered by the assumption that the true image has been globally corrupted by a single blur function.

The aim of this thesis is to develop fast image restoration methods which provide better quality deblurring and give fast and robust results in the blind, non-blind and semi-blind cases. We develop new models to achieve this aim and present experimental results demonstrating their effectiveness.

We begin with a review of some preliminary mathematics in Chapter 2 which may be useful during the reading of this thesis. We then present some existing work in Chapter 3 which is relevant to the work presented in later chapters. We next present the application of some of the ideas introduced in Chapter 3 with some refinements before moving onto the main work of this thesis in Chapters 5–8 which deals with the implicit application of optimisation constraints to improve non-blind and blind deconvolution. We also consider convex relaxation, obtaining improved solution speeds, improving deblurring approximations by separating noise and formulating parametric approximations of piece-wise constant functions. Finally, we present an application of this work to the segmentation of blurred images.

List of Figures

List of Tables

List of Algorithms

Publications

Mathematical Methods and Algorithms for Restoring Retinal Images with Blur and Blind Blur. Bryan M. Williams, Ke Chen, Yalin Zheng and Simon P. Harding. *Ophthalmology Image Analysis Workshop, University of Liverpool.* December 02, 2011.

Advanced Blur Removal Methods with Applications to Retinal Imaging for Ophthalmology. Bryan M. Williams, Ke Chen, Yalin Zheng and Simon P. Harding. *Medical Image Understanding and Analysis 2012.* 286–291. 2012.

A New Constrained Total Variational Deblurring Model and Its Fast Algorithm. Bryan M. Williams, Simon P. Harding and Ke Chen. *Numerical Algorithms.* To appear. 2014.

A New Study of Blind Deconvolution with Implicit Incorporation of Non-negativity Constraints. Bryan M. Williams, Ke Chen, Yalin Zheng and Simon P. Harding. *International Journal of Computational Mathematics.* To appear. 2014.

An Effective Variational Model for Simultaneous Reconstruction and Segmentation of Blurred Images. Bryan M. Williams, Jack A. Spencer, Ke Chen, Yalin Zheng and Simon P. Harding. *Computer Vision Image and Understanding Special Issue on Discrete and Continuous Geometry in Computer Vision.* Submitted. 2014.

An Enhanced Model for Semi-Blind Deblurring with Parametric Kernel Identification and Constrained Image Reconstruction. Bryan M. Williams, Ke Chen, Yalin Zheng and Simon P. Harding. In Preparation. 2014.

Presentations

Mathematical Methods and Algorithms for Restoring Retinal Images with Blur and Blind Blur.
Bryan M. Williams. Ophthalmic Image Analysis Workshop, University of Liverpool. December 02, 2011.

Mathematical Methods for Blind Image Deblurring.
Bryan M. Williams. DITANET Symposium, Cockcroft Institute. May 16, 2012.

A Blind Deconvolution Model for Images Corrupted by Gaussian Blur.
Bryan M. Williams. Essex-Greenwich-Hertfordshire Workshop on Applied and Numerical Mathematics - Multiscale Problems, University of Greenwich. June 07–08, 2012.

Advanced Blur Removal Methods with Applications to Retinal Imaging for Ophthalmology.
Bryan M. Williams. 16th Conference on Medical Image Understanding and Analysis, Swansea University. July 09–11, 2012.

Advanced Mathematical Methods for Medical Image Enhancement.
Bryan M. Williams. Workshop on Ageing, University of Liverpool. December 19, 2012.

Mathematical Deblurring of Images for Non-Blind and Blind Restoration.
Bryan M. Williams. 25th Biennial Numerical Analysis Conference, University of Strathclyde. June 24–28, 2013.

Image Deconvolution Techniques.
Bryan M. Williams. LMS Inverse Problems Meeting - Tomographic Reconstructions from Boundary Data, University of Leeds. September 22, 2014. Awarded prize for best poster.

Chapter 1

Introduction

This thesis is about developing mathematical models for image restoration, particularly in image deblurring.

In this chapter, we present an introduction to deblurring as an image processing technique in §1.1 along with a brief overview of how the problem of blur degradation can be modelled and the different classes of image deblurring problems encountered. We also present in §1.2 an outline of the chapters of this thesis.

1.1 Introduction to Image Deblurring

Image processing incorporates many problems such as image reconstruction which includes removing image noise from a given image by *denoising* [26, 28, 56, 169], the task of reconstructing an image from a given blurred image known as *deblurring* [54, 123, 177, 104, 2, 208, 210] and reconstructing a missing or damaged portion of an image (*inpainting*) [6, 16, 25, 52]. Other important image processing techniques include emphasising the boundaries of an image by different filters or *segmenting* an image into subregions [8, 12, 53, 69, 162] and *registration* which attempts to align two images based on some measure of similarity [171, 139, 140, 106, 110, 189, 77] as well as others [107, 130, 146, 73, 217, 80, 128, 159, 101, 158, 175].

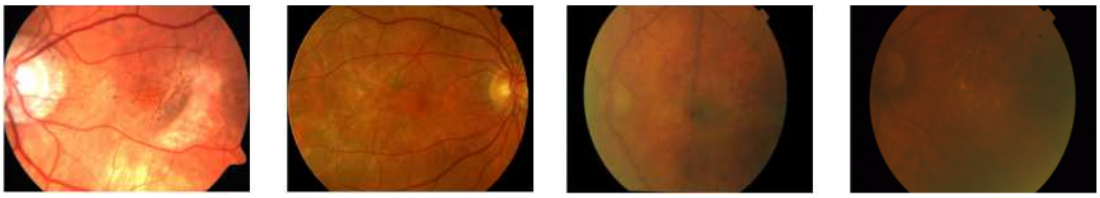
A significant problem in image processing is *image deblurring* or *deconvolution*. This is the process of restoring a hidden true image from a given blurred image enabling edges to be sharper, hidden features to be revealed and details to be clearly visible. In this section we introduce the issue of blur in images and the task of restoring an image from blur degradation.

1.1.1 The Problem of Blur Degradation of Images

Blurring of images occurs in many fields such as astronomical imaging, remote sensing, microscopy and medical imaging such as Colour Fundus Angiography for retinal imaging, which is our main interest.

Despite many technological advances in retinal imaging, blurring remains a major cause for image quality degradation in clinical settings where accurate scans are diffi-

cult to obtain yet vital for diagnosis and screening as well as further processing such as registration. It is particularly important to get as much information as possible of retinal vessels and other structures for treatment and management as well as for grading. Some examples of retinal scans which vary in quality from excellent to inadequate are shown in Figure 1.1.



Excellent Good Fair Inadequate

Figure 1.1: Examples of Colour Fundus images of varying quality from *excellent* where blood vessels and other details can clearly be seen to *inadequate* where most of the detail is not visible. Such inadequate images cannot be used for diagnosis or screening.

Blurring in retinal images leads to a substantial number of unnecessary referrals and in turn a waste of valuable hospital resource. In a current programme which sees three million diabetic patients undergo annual photographic screening, approximately 10% of the images acquired (by high-resolution digital cameras) are considered to be too blurred for assessment and so ungradeable due to inadequate clarity or poor field definition. This proportion of inadequate scans is typical of retinal imaging and may result in further referrals or even misdiagnosis. Such visually ungradeable images are more likely to come from patients who have reached an advanced stage of Retinopathy.

Blurring of images is due to many factors such as motion of the camera or the target scene, defocusing of the lens system, imperfections in the electronic, photographic, transmission medium, and obstructions. In retinal imaging, there are many contributing factors influencing the quality of the received scan including patient-related factors such as eye movement and the age of the patient. Those who are particularly young or old find it difficult to keep the eye still during the process, making it difficult to obtain an adequate scan. Advanced ocular diseases and other coexisting conditions such as Parkinson's disease also make it difficult for light to pass through the eye and can cause blur. Refractive error, difficulty maintaining careful focus and the skill and experience of the photographer are also contributing factors.

1.1.2 Modelling the Blurring Process

An observed blurred image can be written as a convolution of the true image with a linear shift-invariant (lsi) blur, known as the *point spread function* (psf) or unknown kernel function $\kappa(x, y)$ [117].

In order to model the blurring of an image $u(x, y)$, we discretise the image function over an $n_x \times n_y$ pixel mesh. Letting $z(x, y)$ denote the received (corrupted) image, $\kappa(x, y)$ denote the blur function, $\eta(x, y)$ denote noise which is introduced during the

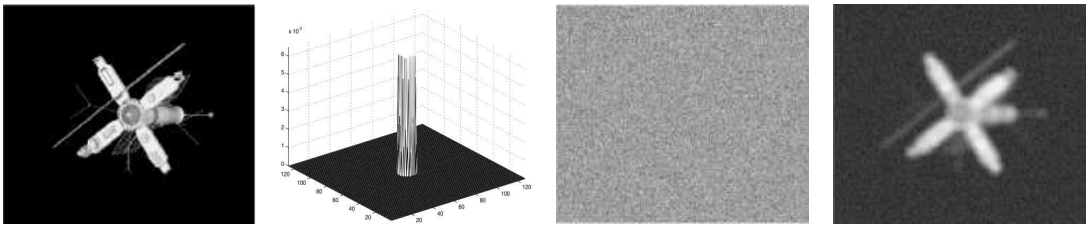
collection of data and $u(x, y)$ denote the hidden true image which we wish to recover, we model the blurred image $z(x, y)$ as

$$z(x, y) = [\kappa * u](x, y) + \eta(x, y)$$

where $*$ denotes the operation of convolution which is given by

$$[\kappa * u](x, y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \kappa(x - x', y - y') u(x', y') \, \mathrm{d}x' \, \mathrm{d}y'$$

for the convolution of the kernel $\kappa(x, y)$ and the true image $u(x, y)$. See Figure 1.2 for an example of forming a blurred and noisy image from a sharp, clean image.



(a) True Image $u(\boldsymbol{x})$ (b) Blur Function $\kappa(\boldsymbol{x})$ (c) Noise Function $\eta(\boldsymbol{x})$ (d) Blurred Image $z(\boldsymbol{x})$

Figure 1.2: Illustration of the blurring process. From left to right we have (a) the true image $u(\boldsymbol{x})$, (b) the blur function $\kappa(\boldsymbol{x})$, (c) the additive noise acquired $\eta(\boldsymbol{x})$, and (d) the received image $z(\boldsymbol{x}) = [\kappa * u](\boldsymbol{x}) + \eta(\boldsymbol{x})$.

1.1.3 Restoring Images from Blur Degradation

Attempting to recover the lost true image $u(x, y)$ typically leads to solving the large system

$$A\boldsymbol{u} = \boldsymbol{b} \qquad \text{where} \qquad \boldsymbol{b} = K^\top \boldsymbol{z}, \qquad A = K^\top K + \alpha L,$$

where K is an ill-conditioned (typically full) matrix, $\boldsymbol{z}$ is the received corrupted data, L is a sparse regularisation matrix which may or may not be symmetric, and α is scalar quantity known as the regularisation parameter, usually small and positive, which is used to control the amount of regularisation.

There are three main deconvolution problems:

1. Deconvolution in the case of known blur, known as *non-blind deconvolution*, has been investigated widely in the last few decades giving rise to a variety of solutions [119, 178, 11, 207, 13, 94, 104, 193]. In this case, the point spread function is assumed known even though this information is not available in most of the real applications. The challenge is to recover the true image.

2. *Semi-blind deblurring* involves problems for which some important information about the blur kernel is assumed or known [11, 14, 58, 144]. For example, the blur function may be assumed to belong to a class of parametric functions, such

as a Gaussian function. The task then is to recover the image and estimate the correct parameters which identify the blur.

3. *Blind image deconvolution* includes the cases where both the blur kernel and the image are unknown [216, 117, 118, 54, 55, 141, 2, 31, 64, 78, 103, 126]. In these cases, the problem becomes harder and much more challenging since both the blur function and true image must be found, but it has much greater applicability to real-world problems.

Each of the above types of deblurring problem is important in many scientific applications such as astronomical imaging, medical imaging, and remote sensing as well as consumer photography.

1.2 Thesis Outline

The remaining chapters of this thesis are organised as follows.

Chapter 2 - Mathematical Preliminaries

In this chapter, we present some mathematical tools which will be used throughout this thesis which the reader may wish to review while reading subsequent chapters. A brief review will be given with definitions, theorems and examples of some important relevant mathematical topics including linear spaces, variations of a functional, bounded space of variation, inverse problems and image representation. The discretisation of partial differential equations using finite differences and iterative solutions of linear and non-linear equations are also presented. An overview of some relevant numerical methods such as Newton's method and the Steepest Descent method will be given as well as some fast solver algorithms including Conjugate Gradient.

Chapter 3 - Review of Variational Models for Image Reconstruction

In this chapter, we present a brief revision of some variational models for image restoration and reconstruction techniques. We begin with an introduction to noise corruption in images and denoising techniques using filters such as the total variation (tv) regularisation functional based on the Rudin-Osher-Fatemi (ROF) denoising model, introducing some its properties and covering the benefits and drawbacks before discussing alternative forms of regularisation. We then introduce the problem of blur corruption in images in more detail and introduce some deconvolution techniques in the non-blind, semi-blind and blind cases. We also present some work in image segmentation which may be useful when reading Chapter 8.

Chapter 4 - Application to Blurred Images and some Refinements

In this chapter, we present the application of some of the image deblurring models presented in Chapter 3. We derive the Euler-Lagrange equations which may be solved

to minimise the functional and discuss the methods of solving such equations accurately and efficiently. We then present some refinements to these models. We consider an enhancement of such models achieved by varying the boundary conditions in the cases of blind and non-blind deconvolution and a fixed-point method for solving the resulting system quickly, presenting experimental results. We next present a method for obtaining a fast solution to the deblurring problem by variable splitting, separating a dense and non-linear system into two systems which may each be solved efficiently. Finally, we present a method which aims to improve blur function estimation in the case of blind deblurring by imposing filtering on the kernel function and considering methods of regularisation depending on assumption of the blur type.

Chapter 5 - A New Constrained Deblurring Model

In this chapter we consider techniques for preserving non-negativity and in particular the intensity range in image deblurring and discuss the introduction of non-negativity constraints both explicitly and implicitly. Beginning with a constrained model, we develop an unconstrained model which is able to successfully preserve the range of intensity values of an image when it is reconstructed from a corrupt image using known information about the blurring function. We consider alternative regularisation and two forms of the constrained model given by linearisation and by alternate direction methods. We then discuss convexity of the model, introducing a convex variant of the functional. Numerical evidence showing the quality of reconstruction is presented.

Chapter 6 - A Robust Model for Constrained Blind Image Deblurring

In this chapter, we review some methods for solving the constrained blind deconvolution problem and the issues of imposing such artificial constraints and of relaxing them. By imposing such constraints, we may not minimise the given function, but by not imposing the non-negativity constraint or the upper-value constraint, we risk converging towards a false solution. We present an adaptation of the implicitly constrained deblurring model for use when identifying the cause of the blur corruption. In this case, we assume that the blur function satisfies a non-negativity constraint. We present a model to impose this constraint implicitly and a strategy for dealing with the non-linearity and recovering the image and blurring kernel simultaneously. We next present a modification of this method which is able to obtain a fast result and incorporates alternative regularisation for smooth blur functions. Finally, we present experimental results demonstrating the effectiveness of this work.

Chapter 7 - Semi-Blind Deblurring with Parametric Kernel Identification

In this chapter, we introduce the idea of parametric deblurring and give a brief review of current models which aim to restore an image which has been corrupted by blur given the assumption that the blur is of a particular type, such as Gaussian or out of

focus. We present a parametric deblurring model for images corrupted by Gaussian blur which may be modelled using a well-known formulation. We next present three enhancements to this model by implicitly constraining the range of the image intensity values, considering regularisation of the blur function and separation of the image noise in the energy functional. We then present formulations for alternative blur types (out of focus, linear motion and box blurs) which are able to well approximate these piecewise-constant blur functions. These formulations are continuous but give a very close approximation to the blur kernels such that the difference error, particularly after discretisation, is extremely small. We thus present models which incorporate the new formulation of these blur functions in the functional with experimental results.

Chapter 8 - Simultaneous Reconstruction and Segmentation of Blurred Images

In this chapter, we introduce an application of image deblurring to a method for segmenting blurred images. We present a review of some existing work in this area and a model for segmenting images which have been corrupted by blur and Poisson noise in a two-stage framework which involves first approximating the sharp image and then applying a segmentation technique to the result which aims to find the global minimum of the segmentation problem, which is typically non-convex. We then introduce a new joint model which aims to simultaneously segment the image as it is restored from blur degradation, followed by an accelerated model which uses alternate direction methods to connect the problems of segmentation and deblurring in order to form a joint energy functional. We present some relevant measures of error, including four novel measures and experimental results which demonstrate the effectiveness of this work.

Chapter 9 - Conclusions and Future Research

In the final chapter we present conclusions of the work presented in this thesis and propose possible future research directions arising from the work presented in this thesis.

Chapter 2

Mathematical Preliminaries

In this chapter, we present some mathematical theory which may be useful to the reader throughout this thesis. We begin with an introduction to normed linear spaces with some useful examples and present some relevant theory about calculus of variation. Next, we discuss inverse problems and regularisation before moving on to image representation and discretisation, finishing with an introduction to some iterative methods for solving partial differential equations.

2.1 Normed Linear Spaces

In order to extend working on the real line $\mathbb{R}$ to n-dimensional Euclidean space $\mathbb{R}^n$, we require some structure. This leads to the class of normed linear spaces.

We first introduce some relevant notation and definitions which will be useful throughout this thesis and which may found in literature relating to advanced calculus and linear algebra.

We represent the elements $\boldsymbol{x} \in \mathbb{R}^n$ of the n-dimensional Euclidean space by n-tuples $(x_1, \ldots, x_n)$, called vectors or points in $\mathbb{R}^n$ which for brevity we will usually denote with a bold character, such as

$$\boldsymbol{x} \;=\; (x_1, \ldots, x_n) \;\in\; \mathbb{R}^n. \tag{2.1}$$

The structures of the *inner product* or *dot product* of two vectors $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$ is defined as

$$\boldsymbol{x} \cdot \boldsymbol{y} \;=\; \langle \boldsymbol{x}, \boldsymbol{y} \rangle \;=\; x_1 y_1 + \cdots + x_n y_n, \tag{2.2}$$

and the *length* of a vector as

$$\|\boldsymbol{x}\| \;=\; \sqrt{\boldsymbol{x} \cdot \boldsymbol{x}} \;=\; \sqrt{x_1^2 + \cdots + x_n^2}, \tag{2.3}$$

also known as *Euclidean norm* or *magnitude*. More generally, a norm is a function which maps each non-zero vector of a vector space to a strictly positive real number, satisfying certain properties. More formally, the concept of a norm can be generalised with the following definition.

Definition 2.1.1 (**Norm**). *Let $N : L \subseteq \mathbb{R}^n \to \mathbb{R}$ be a real valued function. Then N is called a* norm *on L if it satisfies the following properties for all $\boldsymbol{x}, \boldsymbol{y} \in L$:*

- $N(\boldsymbol{x}) = 0 \Rightarrow \boldsymbol{x} = \boldsymbol{0}$,
- *Absolute homogeneity:* $N(\alpha \boldsymbol{x}) = |\alpha| N(\boldsymbol{x}) \ \forall \, \alpha \in \mathbb{R}$,
- *Triangle inequality:* $N(\boldsymbol{x} + \boldsymbol{y}) \leq N(\boldsymbol{x}) + N(\boldsymbol{y})$.

The norm of a vector $\boldsymbol{x}$ is commonly represented by $\|\boldsymbol{x}\|$.

Remark 2.1.2 *By the absolute homogeneity axiom, we have $N(-\boldsymbol{x}) = N(\boldsymbol{x})$ so that by the triangle inequality, we have positivity, i.e. $N(\boldsymbol{x}) \geq 0$.*

Remark 2.1.3 *The definition of a* seminorm *is the same as that of a norm but without the first axiom.*

Some examples of functions which are norms are given below.

Example 2.1.4 (**p-norm**). *Let $\boldsymbol{x} \in \mathbb{R}^n$, then for any $p \in \mathbb{R}_{\geq 1}$ we define the p-norm of $\boldsymbol{x}$ as*

$$\|\boldsymbol{x}\|_p = \left(\sum_{i=1}^{n} |x_i|^p \right)^{1/p}. \tag{2.4}$$

Note that the Euclidean norm given above in Equation (2.3) can be achieved by setting $p = 2$ in (2.4).

Example 2.1.5 (L^p**-norm**). *Let f be a function defined on a domain Ω and $1 \leq p \leq \infty$. We define the L^p-norm of f on Ω as*

$$\|f(x)\|_p = \left(\int_\Omega |f(x)|^p \ \mathrm{d}x \right)^{1/p}. \tag{2.5}$$

Note that since f may have arbitrarily many components, this is a generalisation of Example 2.1.4.

Example 2.1.6 (L^∞**-norm**). *The special case of the L^p-norm from Example 2.1.5 where $p = \infty$ is defined as*

$$\|f(x)\|_\infty = \sup_x |f(x)|. \tag{2.6}$$

We may now define the concepts of linear spaces and in particular normed linear linear spaces.

Definition 2.1.7 (**Linear space**). *Let $L = (P, K, I)$ be an incidence structure. That is, a triple where P is a set of points, L is a set of lines and $I \subseteq P \times K$ is an incidence relation such that if $(p, l) \in I$ then we consider the point p to lie on the line l. Then L is a linear space if*

- *Two points are incident with exactly one line. That is, if $p_1, p_2 \in P$ and $l_1, l_2 \in K$ are distinct points and lines such that*

$$\{(p_1, l_1), (p_2, l_1), (p_1, l_2), (p_2, l_2)\} \subseteq I$$

then $l_1 = l_2$.

- *Each line is incident with at least two points. That is, for all $l \in K$ there exist distinct $p_1, p_2 \in P$ such that $(p_1, l) \in I$ and $(p_2, l) \in I$.*

Definition 2.1.8 (**Normed linear space**). *A normed linear space is a pair (L, N) where L is a linear space and N is a norm defined on L.*

Definition 2.1.9 (**Cauchy sequence**). *Let $\{\boldsymbol{x}_i\}$ be a sequence in the space L. We say that this is a Cauchy sequence if for every $\varepsilon \in \mathbb{R}_{>0}$ there exists $M \in \mathbb{Z}^+$ such that for every natural number $m, n > M$, we have $\|\boldsymbol{x}_m - \boldsymbol{x}_n\| < \varepsilon$.*

Definition 2.1.10 (**Banach space**). *A normed linear space L is called a* Banach Space *if it is complete. That is, if every Cauchy sequence in L converges to an element in L.*

Remark 2.1.11 *For an example of a Banach space, we may take any Hilbert space, defined as an inner product space which is also a metric space with respect to the distance function induced by the inner product. Euclidean space is an example of a Hilbert space.*

2.1.1 Convex Functions

Here, we introduce the idea of convex functions and present some examples.

Definition 2.1.12 (**Linear transformation**). *A function $T : L \to M$ is called a* linear transformation *if, for any $\boldsymbol{x}, \boldsymbol{y} \in L$ and any $\alpha \in \mathbb{R}$, we have*

1. $T(\boldsymbol{x} + \boldsymbol{y}) = T(\boldsymbol{x}) + T(\boldsymbol{y})$,
2. $T(\alpha \boldsymbol{x}) = \alpha T(\boldsymbol{x})$.

Definition 2.1.13 (**Convex function**). *Let S be a convex subset of an n-dimensional vector space V, that is for any $r > 1$ vectors $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_r \in S$ and any $\lambda_1, \ldots, \lambda_r \in \mathbb{R}_{\geq 0}$ such that $\lambda_1 + \ldots + \lambda_r = 1$ we have $\lambda_1 \boldsymbol{x}_1 + \ldots + \lambda_r \boldsymbol{x}_r \in S$. Then a function f defined on S is called* convex *if for all $\boldsymbol{x}_i, \boldsymbol{x}_j \in S$ and $\alpha \in (0, 1)$ we have*

$$f(\alpha \boldsymbol{x}_i + (1 - \alpha)\boldsymbol{x}_j) \leq \alpha f(\boldsymbol{x}_i) + (1 - \alpha) f(\boldsymbol{x}_j). \tag{2.7}$$

f is called strictly convex *if the inequality is strict for $\boldsymbol{x}_i \neq \boldsymbol{x}_j$.*

Remark 2.1.14 *There are several operations which preserve convexity:*

- *Weighted sums: Let f and g be convex functions on $\mathbb{R}^n$. Then the linear combination $h = \alpha f + \beta g$ is also convex for $\alpha, \beta \geq 0$.*
- *Affine substitutions of the argument: Let f be a convex function on $\mathbb{R}^n$ and $\mathcal{P} : \mathbb{R}^m \to \mathbb{R}^n$ be an affine mapping given by $\mathcal{P}(x) = Ax + b$. Then $f(\mathcal{P}(x))$ is also convex.*
- *Pointwise sup: Let f_α define a family of convex functions on $\mathbb{R}^n$. Then the upper bound $\sup_\alpha f_\alpha$ is convex.*

In all cases, the statement holds if the resulting function is finite at least at one point.

Theorem 2.1.15 *Let $\mathcal{A} = (a, b)$ be an interval on $\mathbb{R}$. Then*

1. *A function f which is differentiable everywhere on $\mathcal{A}$ is convex on $\mathcal{A}$ if and only if its derivative is monotonically non-decreasing on $\mathcal{A}$.*

2. *A function f which is twice differentiable everywhere on $\mathcal{A}$ is convex on $\mathcal{A}$ if and only if its second derivative is non-negative on $\mathcal{A}$.*

Example 2.1.16 *Let $u : \Omega \subseteq \mathbb{R}^2 \to \mathbb{R}$ and define the total variation $TV(u)$ of u as*

$$TV(u) = \int_\Omega |\nabla u|.^1$$

We show that the total variation is a convex function as follows. Let $u_1 \neq u_2$ be two functions. Then

$$TV(\alpha u_1 + (1-\alpha)u_2) = \int_\Omega |\nabla(\alpha u_1 + (1-\alpha)u_2)| = \int_\Omega |\alpha \nabla u_1 + (1-\alpha)\nabla u_2|$$

$$\leq \alpha \int_\Omega |\nabla u_1| + (1-\alpha)\int_\Omega |\nabla u_2| = \alpha TV(u_1) + (1-\alpha)TV(u_2).$$

Example 2.1.17 *The square of the L^2-norm of a function $u : \Omega \subseteq \mathbb{R} \to \mathbb{R}$ given by*

$$\|u\|_2^2 = \int_\Omega |u|^2 \,\mathrm{d}x$$

is convex. We show this as follows. Introducing a function ϕ and parameter ε, we can calculate the second derivative of a function $F(u)$ by making the substitution $v = u + \varepsilon\phi$ and finding the second derivative with respect to ε as follows:

$$\frac{\mathrm{d}^2 F(v)}{\mathrm{d}\varepsilon^2} = \frac{\mathrm{d}}{\mathrm{d}\varepsilon}\left(\frac{\mathrm{d}F(v)}{\mathrm{d}v}\frac{\mathrm{d}v}{\mathrm{d}\varepsilon}\right) = \frac{\mathrm{d}}{\mathrm{d}\varepsilon}\left(\frac{\mathrm{d}F(v)}{\mathrm{d}v}\phi\right) = \frac{\mathrm{d}^2 F(v)}{\mathrm{d}v^2}\frac{\mathrm{d}v}{\mathrm{d}\varepsilon}\phi = \frac{\mathrm{d}^2 F(v)}{\mathrm{d}v^2}\phi^2.$$

Applying this idea to the L^2-norm, we have

$$\frac{\mathrm{d}^2}{\mathrm{d}\varepsilon^2}\|u+\varepsilon\phi\|_2^2 = \int_\Omega \frac{\mathrm{d}^2}{\mathrm{d}\varepsilon^2}(u+\varepsilon\phi)^2 \,\mathrm{d}x = \int_\Omega \frac{\mathrm{d}}{\mathrm{d}\varepsilon} 2(u+\varepsilon\phi)\phi \,\mathrm{d}x = 2\int_\Omega \phi^2 \,\mathrm{d}x$$

and so the second derivative is non-negative. Then by Theorem 2.1.15, the square of the L^2-norm of the function u is convex.

Note also that by Remark 2.1.14, the square of the L^2-norm of a linear combination of functions given by

$$\|u + \lambda v\|_2^2 = \int_\Omega |u + \lambda v|^2 \,\mathrm{d}x$$

is also convex for functions $u, v : \Omega \subseteq \mathbb{R} \to \mathbb{R}$ and $\lambda \in \mathbb{R}$.

2.1.2 Differentiable Functions

We would now like to generalise the idea of a derivative of a real-valued function of a single variable to the case of a vector valued function of multiple variables. The Fréchet derivative generalises this idea to functions on Banach spaces.

[1] Note that the total variation $\int_\Omega |\nabla u(\boldsymbol{x})|$ is commonly written as $\int_\Omega |\nabla u(\boldsymbol{x})| \,\mathrm{d}\boldsymbol{x}$ in the literature.

Definition 2.1.18 (**Fréchet derivative**). *Let $f : U \to W$ be a function defined on an open subset $U \subseteq L$ of a Banach space L, taking values in a second Banach space W. Then we say that f is Fréchet differentiable at $\boldsymbol{x} \in U$ if there exists a bounded linear operator $T_{\boldsymbol{x}} : L \to W$ such that*

$$\lim_{\boldsymbol{h}\to\boldsymbol{0}} \frac{\|f(\boldsymbol{x}+\boldsymbol{h}) - f(\boldsymbol{x}) - T_{\boldsymbol{x}}(\boldsymbol{h})\|_W}{\|\boldsymbol{h}\|_L} = 0.$$

We then call $f'(\boldsymbol{x}) = T_{\boldsymbol{x}}$ the Fréchet derivative of f at $\boldsymbol{x}$.

In contrast, the Gâteau derivative generalises the idea of the directional derivative.

Definition 2.1.19 (**Gâteux derivative**). *Let f be a function defined on an open subset $U \subseteq L$ of a Banach space L, taking values in a second Banach space W. Then we say that $f : U \to W$ is Gâteau differentiable at $\boldsymbol{x} \in U$ if the two-sided directional derivative $f'(\boldsymbol{x}_0; \boldsymbol{v})$ exists for each $\boldsymbol{v} \in L$; that is, if there exists $g : V \to W$ such that*

$$g(v) = \lim_{\varepsilon\to 0} \frac{f(\boldsymbol{x}+\varepsilon\boldsymbol{v}) - f(\boldsymbol{x})}{\varepsilon}. \tag{2.8}$$

Remark 2.1.20 *Note that if the Fréchet derivative of f exists at $\boldsymbol{x}$, then f is Gâteaux differentiable at $\boldsymbol{x}$ and the two derivatives coincide but the opposite does not hold.*

2.2 Calculus of Variations

We now introduce the basic theory which we require in order to derive the first variation or Euler-Lagrange equation of a functional. The reader may wish to consult sources such as [81, 83, 84] for more information.

We begin by introducing the first variation of a functional and a necessary condition to be satisfied by a minimiser of a variational integral. We then present the Divergence theorem, leading to integration by parts and some relevant examples of finding the first variation of a functional.

2.2.1 Variation of a Functional

Let

$$J[u] = \int_\Omega F(\boldsymbol{x}, u(\boldsymbol{x}), \nabla u(\boldsymbol{x}))\ \mathrm{d}\boldsymbol{x} \tag{2.9}$$

be a functional which is defined on some normed linear space and dependent on an independent variable $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)$, an unknown function $u(\boldsymbol{x})$, and its gradient given by $\nabla u(\boldsymbol{x}) = (u(\boldsymbol{x})_{x_1}, u(\boldsymbol{x})_{x_2}, \ldots, u(\boldsymbol{x})_{x_n})$. Here $\mathrm{d}\boldsymbol{x}$ is the n-differential element defined as $\mathrm{d}\boldsymbol{x} = \mathrm{d}x_1\,\mathrm{d}x_2 \cdots \mathrm{d}x_n$.

The most important necessary condition to be satisfied by any minimiser of a variational integral like $J(u)$ is the vanishing of its first variation δJ defined as

$$\delta J(u) = \frac{\mathrm{d}}{\mathrm{d}\varepsilon} J(u+\varepsilon\varphi)\bigg|_{\varepsilon=0}. \tag{2.10}$$

This is, if u is a minimiser of $J(u)$ with respect to variations $\delta u = \varphi$ which do not change boundary values of u, then (2.10) must be satisfied for all φ with compact support[2] in Ω. Then for some $u_0 \in \Omega$, we call $\delta J(u_0)$ the first variation of J at u_0 in the direction of φ.

To compute the first variation of variational integrals like (2.9) on the direction of a function $\varphi(\boldsymbol{x})$ with compact support in Ω, we define the following transformation:

$$u^*(\boldsymbol{x}) = u(\boldsymbol{x}) + \varepsilon\varphi(\boldsymbol{x}) + O(\varepsilon^2), \tag{2.11}$$

where $\varepsilon \to 0$ and $\|\varphi\| \to 0$. Then the variation δJ of the functional (2.9) corresponding to the above transformation is defined as the linear part in ε of the difference $J[u^*]-J[u]$. Using Taylor's theorem, it is possible to show (see for example [81]) that

$$J[u^*] - J[u] = \varepsilon \int_\Omega \left(F_u + \sum_{i=1}^{n} F_{u_{x_i}} \varphi_{x_i} \right) \mathrm{d}\boldsymbol{x} + O(\varepsilon^2). \tag{2.12}$$

It follows then that the variation of the functional (2.9) is given by

$$\delta J = \int_\Omega \left(F_u + \sum_{i=1}^{n} F_{u_{x_i}} \varphi_{x_i} \right) \mathrm{d}\boldsymbol{x}. \tag{2.13}$$

2.2.2 The Divergence Theorem and Integration by Parts

Consider an open and bounded subset $\Omega \subset \mathbb{R}^n$ with piece-wise smooth boundary $\partial\Omega$. Suppose a scalar function $u(\boldsymbol{x})$ is continuously differentiable on $\bar{\Omega}$. Then

$$\int_\Omega u_{x_i} \,\mathrm{d}\boldsymbol{x} = \int_{\partial\Omega} u(\boldsymbol{x})\nu_i \,\mathrm{d}s \tag{2.14}$$

is satisfied for $i = 1, \ldots, n$ with $\boldsymbol{\nu} = (\nu_1, \nu_2, \ldots, \nu_n)$ the outward unit normal of $\partial\Omega$. Using this result and simple additivity, it is simple to show that for a given vector field $\boldsymbol{F} = \boldsymbol{F}(x)$ the following is true

$$\int_\Omega (\nabla \cdot \boldsymbol{F}) \;\mathrm{d}\boldsymbol{x} = \int_{\partial\Omega} \boldsymbol{F} \cdot \boldsymbol{\nu} \;\mathrm{d}s. \tag{2.15}$$

The latter is the most common way to present the divergence theorem.

An immediate consequence of the divergence theorem is the integration by parts formula. This is, for $i = 1, \ldots, n$ and for two continuously differentiable functions $u(\boldsymbol{x})$ and $v(\boldsymbol{x})$ in $\bar{\Omega}$

$$\int_\Omega u_{x_i} v(\boldsymbol{x}) \;\mathrm{d}\boldsymbol{x} = -\int_\Omega u(\boldsymbol{x}) v_{x_i} \;\mathrm{d}\boldsymbol{x} + \int_{\partial\Omega} u(\boldsymbol{x}) v(\boldsymbol{x}) \nu_i \,\mathrm{d}s. \tag{2.16}$$

Additionally, by applying (2.15) to the product of a scalar function g and a vector

[2]The support of a function is the set of points where this function is not zero. Functions with *compact support* in a space X are those for which their support is a compact subset of X.

field $\boldsymbol{F}$, we obtain the vectorial representation

$$\int_\Omega (\boldsymbol{F}\cdot\nabla g + g\nabla\cdot\boldsymbol{F})\ \mathrm{d}\boldsymbol{x} = \int_{\partial\Omega} g\boldsymbol{F}\cdot\boldsymbol{\nu}\ \mathrm{d}s. \tag{2.17}$$

In order to finalise this short section, we present an example of how to compute the first variation of a functional of our interest.

Example 2.2.1 *Consider the problem of finding the first variation of the functional*

$$F(u) = \int_\Omega |\nabla u|$$

defined on a domain $\Omega\subset\mathbb{R}^2$. To this end, we introduce the small variation $\varepsilon\varphi$ composed of the parameter $\varepsilon\to 0$ and the continuously differentiable function φ with compact support in Ω. Then we compute

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}\varepsilon}F(u+\varepsilon\varphi)\bigg|_{\varepsilon=0} &= \frac{\mathrm{d}}{\mathrm{d}\varepsilon}\int_\Omega |\nabla(u+\varepsilon\varphi)|\bigg|_{\varepsilon=0}\\ = \int_\Omega \frac{\nabla(u+\varepsilon\varphi)}{|\nabla(u+\varepsilon\varphi)|}\cdot\nabla\varphi\bigg|_{\varepsilon=0} &= \int_\Omega \frac{\nabla u}{|\nabla u|}\cdot\nabla\varphi.\end{aligned} \tag{2.18}$$

We can now use integration by parts on (2.18), obtaining

$$\int_\Omega \frac{\nabla u}{|\nabla u|}\cdot\nabla\varphi = -\int_\Omega \nabla\cdot\left(\frac{\nabla u}{|\nabla u|}\right)\varphi + \int_{\partial\Omega}\boldsymbol{\nu}\cdot\nabla u\ \mathrm{d}s \tag{2.19}$$

where $\partial\Omega$ is the boundary of Ω, $\boldsymbol{\nu}$ is the unit outward normal vector to $\partial\Omega$ and $\mathrm{d}s$ is the length element of integration. If we further require

$$\frac{\mathrm{d}}{\mathrm{d}\varepsilon}F(u+\varepsilon\varphi)\bigg|_{\varepsilon=0} = 0,$$

then the following partial differential equation known as the Euler-Lagrange equation must be satisfied:

$$\nabla\cdot\left(\frac{\nabla u}{|\nabla u|}\right) = 0 \ \ \textit{in}\ \ \Omega \tag{2.20}$$

with Neumann boundary condition $\boldsymbol{\nu}\cdot\nabla u = 0$ on $\partial\Omega$.

2.3 Functions of Bounded Variation

In this section, we define the space of functions of bounded variation and present some examples of such functions.

Let Ω be a bounded open subset of $\mathbb{R}^n$ and let $u\in L^1(\Omega)$. Define

$$\begin{aligned}\int_\Omega |Du| = sup\Big\{\int_\Omega u\ \mathrm{div}\boldsymbol{\varphi}\ \mathrm{d}\boldsymbol{x}:\ \boldsymbol{\varphi}=(\varphi_1,\varphi_2,\ldots,\varphi_n)\in\mathcal{C}_0^1(\Omega;\mathbb{R}^n)^n\\ \text{and } |\varphi(\boldsymbol{x})|_{L^\infty(\Omega)}\le 1 \text{ for } \boldsymbol{x}\in\Omega\Big\},\end{aligned} \tag{2.21}$$

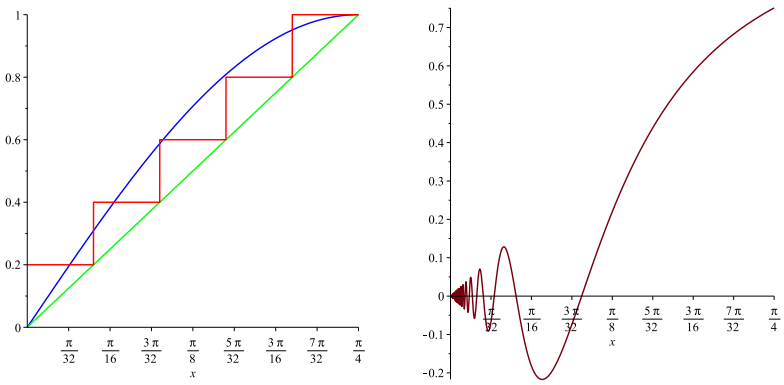


Figure 2.1: On the left, three bounded variation functions with the same total variation. On the right, a function of no bounded variation.

where

$$\mathrm{div}\boldsymbol{\varphi} = \sum_{i=1}^{n} \frac{\partial \varphi_i}{\partial x_i}(\boldsymbol{x}),$$

$\mathrm{d}\boldsymbol{x}$ is the Lebesgue measure[3] and $\mathcal{C}_0^1(\Omega)^n$ is the space of continuously differentiable functions with compact support in Ω. Notice that all the components of $\boldsymbol{\varphi}$ have $L^\infty(\Omega)$-norm less than 1.

A particular and interesting case is when $u \in C^1(\Omega)$, then integration by parts gives

$$\int_\Omega u \ \mathrm{div}\boldsymbol{\varphi} \ \mathrm{d}\boldsymbol{x} = -\int_\Omega \sum_{i=1}^{n} \frac{\partial u}{\partial x_i}\varphi_i \ \mathrm{d}\boldsymbol{x} \tag{2.22}$$

for every $u \in C_0^1(\Omega;\mathbb{R}^n)$, so that

$$\int_\Omega |Du| = \int_\Omega |\nabla u|. \tag{2.23}$$

Definition 2.3.1 (**Function of bounded variation**). *A function $u \in L^1(\Omega)$ is said to have bounded variation in Ω if $\int_\Omega |Du| < \infty$.*

Definition 2.3.2 *We define $BV(\Omega)$ as the space of all functions in $L^1(\Omega)$ with bounded variation.*

Example 2.3.3 *To better understanding bounded variation functions let us briefly analyse some illustrative examples. The following functions $f_1(x), f_2(x)$ and $f_3(x)$ defined below, all belong to $BV(\Omega)$ with, in this case $\Omega = \{x \in [0, \pi/4]\}$:*

$$f_1(x) = \sin(2x), \tag{2.24}$$

[3]In Euclidean spaces, the standard way to assign a *measure* (length, area or volume) to a given subset is through the Lebesgue measure. Hence, sets with finite Lebesgue measure are called Lebesgue measurables. In real analysis, this measure is used to define Lebesgue integration.

$$f_2(x) = \begin{cases} 1/5 & for\, 0 \le x < \pi/20 \\ 2/5 & for\, \pi/8 \le x < \pi/10 \\ 3/5 & for\, \pi/4 \le x < 3\pi/20 \\ 4/5 & for\, \pi/4 \le x < \pi/5 \\ 1 & for\, 3\pi/8 \le x \le \pi/4 \end{cases}, \tag{2.25}$$

$$f_3(x) = \frac{4x}{\pi}. \tag{2.26}$$

What's more, it is not difficult to see, either via direct computation or by just looking at Figure 2.1, that all of the above functions have the same total variation equal to one. However, only $f_1(x)$ and $f_3(x)$ are continuous and differentiable functions on Ω.

Example 2.3.4 *Now consider the function $f_4(x)$ defined as*

$$f_4(x) = \begin{cases} 0 & for\, x = 0 \\ x^2 \sin(1/x) & for\, 0 < x \le a \;\; with \;\; a > 0. \end{cases} \tag{2.27}$$

Here, $\Omega = \{x \in [0, a]\}$ for any $a > 0$. This function is plotted on the right-hand-side in Figure 2.1. We see that as $x \to 0$ the frequency of the oscillations of $f_4(x)$ increases, then the more x approaches zero the more variations need to be added and the value of the integral (2.23) increases. Therefore, this function has infinite total variation and does not belong to $BV(\Omega)$. Notice however that $f_4(x)$ does have bounded variation on $[a, b]$ for any $a > 0$.

Remark 2.3.5 *Under the norm*

$$\|u\|_{BV} = \|u\|_{L^1} + \int_\Omega |Du|, \tag{2.28}$$

$BV(\Omega)$ is a Banach space.

2.4 The Coarea Formula

In this section, we introduce the coarea formula which is very useful for analysis with functions of bounded variation. The coarea function provides a connection between the total variation of a function and the perimeter of its level sets.

Definition 2.4.1 (**Borel set**). *Let X be a topological space and $E \subset X$. We say that E is a Borel set if it can be formed from open sets with a countable number of operations using unions, intersections and the relative complement [170, 20].*

Definition 2.4.2 (**Perimeter**). *Let E be a Borel set and Ω an open set in $\mathbb{R}^n$. We define the perimeter of E in Ω as*

$$P(E, \Omega) = \int_\Omega |D\chi_E| = sup \left\{ \int_E \mathrm{div}\boldsymbol{\varphi} \; \mathrm{d}\boldsymbol{x} : \boldsymbol{\varphi} \in C_0^1(\Omega; \mathbb{R}^n) \; and \; |\boldsymbol{\varphi}(\boldsymbol{x})| \le 1 \right\}, \tag{2.29}$$

where

$$\chi_E = \begin{cases} 1 & if\, \boldsymbol{x} \in E \\ 0 & if\, \boldsymbol{x} \in \Omega \backslash E \end{cases} \tag{2.30}$$

is the indicator function of E.

Definition 2.4.3 (**Coarea formula**). *Let $u = u(\boldsymbol{x})$ and $f = f(\boldsymbol{x})$ be two scalar functions defined on $\mathbb{R}^n$. Assume that u is Lipschitz continuous and that for almost every $\lambda \in \mathbb{R}$, the level set $L = \{\boldsymbol{x} \in \mathbb{R}^n : u(\boldsymbol{x}) = \lambda\}$ is a smooth $(n-1)$-dimensional hypersurface in $\mathbb{R}^n$. Suppose also that f is continuous and integrable. Then*

$$\int_{\mathbb{R}^n} |\nabla u| f \ \mathrm{d}\boldsymbol{x} = \int_{-\infty}^{+\infty} \left(\int_L f \, \mathrm{d}s \right) \ \mathrm{d}\lambda. \tag{2.31}$$

For the particular case when $f = 1$ and the region of integration is a subset $\Omega \subset \mathbb{R}^n$ we have

$$\int_{\Omega} |\nabla u| \ \mathrm{d}\boldsymbol{x} = \int_{-\infty}^{+\infty} \left(\int_L \mathrm{d}s \right) \ \mathrm{d}\lambda = \int_{-\infty}^{+\infty} Per(L, \Omega) \ \mathrm{d}\lambda. \tag{2.32}$$

2.5 Inverse Problems and Regularisation

Inverse problems are commonly encountered in many different branches of science. For instance, water pollution source identification problems [164], hydraulic conductivity identification in steady groundwater flow [74], etc. They are also present in the formulation of many image processing tasks such as denoising [169], deblurring [54], segmentation [53], registration [139] and inpainting [52].

Typically, the variational approach to these problems necessitates some form of regularisation of the inverse problem in order to make it well-posed and hence solvable. In this section, we present the basic theory of inverse problems and regularisation, beginning with the ideas of well and ill-posed problems.

2.5.1 Well and Ill-Posed Problems

From a definition given by Hadamard [90], a *well-posed* problem is one for which:

1. a solution exists,
2. the solution is unique,
3. the solution depends continuously on the initial conditions or data (stability condition).

A problem is considered to be *ill-posed* if any of these criteria are not satisfied.

2.5.2 Inverse Problems

An inverse problem is one where the aim is to recover the model parameters from some known observed data from a physical system. It is considered to be the inverse to the forward problem in which the aim is to compute the data from the model parameters.

Inverse problems are commonly ill-posed, with the violation of the stability condition being frequently the cause.

In the branch of variational models for image restoration it is not uncommon however to find inverse problems where the uniqueness condition fails as well (inpainting for example).

Example 2.5.1 *The classical example to illustrate an inverse problem (linear in this case) is the Fredholm integral equation*

$$f(x) = \int_a^b g(x,y)h(y)\ \mathrm{d}y$$

where the task is to infer $h(y)$ from the data $f(x)$ with smooth $g(x,y)$. Here if the mapping is only injective then the inverse will not be continuous. Hence small errors in the measured data $f(x)$ will introduce large errors in the solution $h(y)$. In this sense the inverse problem is ill-posed.

Example 2.5.2 *We consider now an example from image restoration techniques. A noisy image z defined in $\Omega \subset \mathbb{R}^2$ can easily be constructed by adding a certain quantity of Gaussian noise η to a clean image u in such a way that the relation $z = u+\eta$ is satisfied. Now consider the inverse problem of finding u given only z, implying the removal of the noisy part η. This problem can be approached using variational techniques with the extra assumption that the standard deviation σ^2 of the noise is known or at least can be estimated. In this case, the problem is to find u which minimises*

$$\min_u \left\{ \int_\Omega |u-z|^2\ \mathrm{d}x\,\mathrm{d}y = \sigma^2 \right\}. \tag{2.33}$$

Here, we have an inverse problem that is ill-posed due to (2.33) having many possible solutions.

2.5.3 Regularisation

Regularisation is a technique used to transform an inverse problem into a well-posed problem; Tikhonov *et al.* [190] introduced a popular way to overcome ill-posed minimisation problems. The basic idea is to introduce a new constraint to the problem which demands the solution to belong to a specific set of solutions or to have specific features. For instance, Example 2.5.2 can be *regularised* the following way

$$\min_u \left\{ \int_\Omega |u-z|^2\ \mathrm{d}x\,\mathrm{d}y + \alpha \int_\Omega |\nabla u|^2\ \mathrm{d}x\,\mathrm{d}y \right\}. \tag{2.34}$$

The first term in (2.34) is the *data*, *fidelity* or *fitting term* and the second is the *regularisation term* which requires the solution u to have low gradient values thereby removing noise. α is a tuning parameter which measures the trade-off between regularisation and data fitting. This parameter can be selected manually or empirically, although there exists recent research dedicated to finding methods of selecting this parameter automatically for certain applications.

Other examples of the use of Tikhonov regularisation can be found in [136, 68, 193] and the references therein. More examples of regularisation will be seen in Chapter 3 and throughout this thesis.

2.5.4 Regularisation Parameter Selection

There are several known methods for selecting the regularisation parameter automatically. The aim is to automatically select the value of the parameter which gives a good quality restored image. We present two of those methods here.

L-curve method

This method involves solving the Tikhonov regularised problem (2.34) for a particular set of α_i, obtaining $u(\alpha_i)$. We then plot the graph given by

$$(x(\alpha_i), y(\alpha_i)) = \left(\log\left(||u(\alpha_i) - z||_2^2\right), \log\left(||u(\alpha_i)||_2^2\right)\right),$$

which gives an L-shaped graph. In order to give a better choice of parameter, we select the value at the corner of the L. See [91, 193] for more details.

2.6 Image Representation

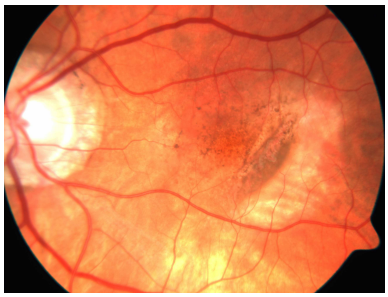
In this section, we look at some of the most commonly used methods of representing images.

2.6.1 Computational Representation

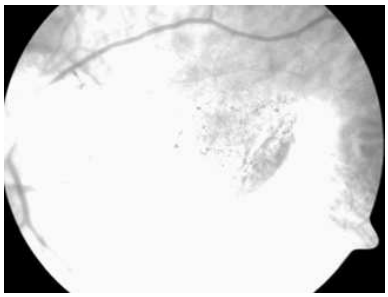
From a computation point of view, an image is an array which may be 2-dimensional or of higher dimension. The simpler case is that of grey-scale images, which may be viewed as a collection of values contained in a 2-dimensional array. Such a value is given at each *pixel* of the image and represents the level of brightness or intensity at that point. These values will be referred to as *intensity values*. Intensity values typically lie in the range $[0, 255]$ or, if the image has been normalised, $[0, 1]$ or $[0, m]$ where $m = \sum\sum u_{i,j}$. The third option is typically reserved for certain functions defined over the image domain and allows us to assume that the integral over the domain is equal to one. The upper limit of the intensity values of an image is often given by the bits-per-sample rate b such that the intensity values may lie in the range $\left[0, 2^b - 1\right]$; In the case of colour or multi-channel images, the image is represented as a higher-dimensional array given by

$$U = [u_{i,j,k}]_{m \times n \times p}$$

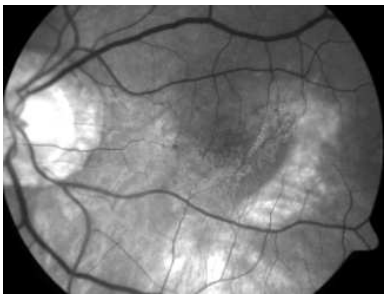
for an $m \times n$-pixel image. Here, p represents the number of channels of the image. For example, representing U in the RGB colour scheme, we have $U = [u_{i,j,k}]_{m \times n \times 3}$. At an arbitrary pixel of the image, the intensity value is a vector $u_{i,j} = (u_{i,j,1}, u_{i,j,2}, u_{i,j,3})$ where each of the entries represent the intensity level at that pixel from each of the channels red (R), green (G) and blue (B). CMYK and certain other representations can be considered similarly.



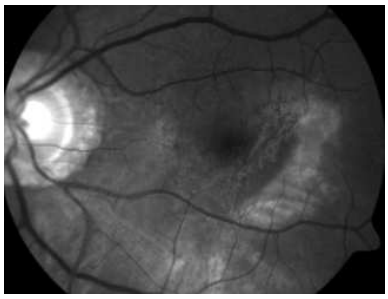
(a) RGB Retina Image u^{RGB}



(b) Channel 1 (Red) u^R



(c) Channel 2 (Green) u^G



(d) Channel 3 (Blue) u^B

Figure 2.2: Illustration of the RGB channels of a Colour Fundus retina image. Note that much information can be seen in the green channel (c).

2.6.2 Mathematical Representation

To consider a grey-scaled image mathematically, we consider it as a smooth function $u : \Omega \to \mathbb{R}$ whose domain is given by a subset $\Omega \subset \mathbb{R}^2$. In this way, we may view images as parametrised curves or by considering the intensity value as the value of the vertical axis (See Figure 2.3).

For this thesis, we see the image primarily as a surface. Regarding the image as a 3D surface, we characterise it by $z = u(x, y)$ where z is the intensity value at each point. The image surface is then defined as $(x, y, u(x, y))$. Information regarding viewing an image as a parametrised curve can be found in [7].

2.7 Discretisation of Partial Differential Equations

The necessity to solve a discrete version of a continuous partial differential equation (pde) arises in many situations, often because the equation cannot be solved analytically or because only discrete data is available.

There are many ways to address this issue, such as the finite element method. For image processing tasks, the domain $\Omega \subset \mathbb{R}^2$ is typically rectangular and the values of a given function are known at uniformly distributed points in the domain. It is therefore common to use the *finite difference method* to discretise the domain.

We proceed by considering a two-dimensional problem with domain $\Omega = (a, b) \times (c, d)$ for $a, b, c, d \in \mathbb{R}$. We impose an $(n_x + 1) \times (n_y + 1)$ Cartesian grid (or mesh) with

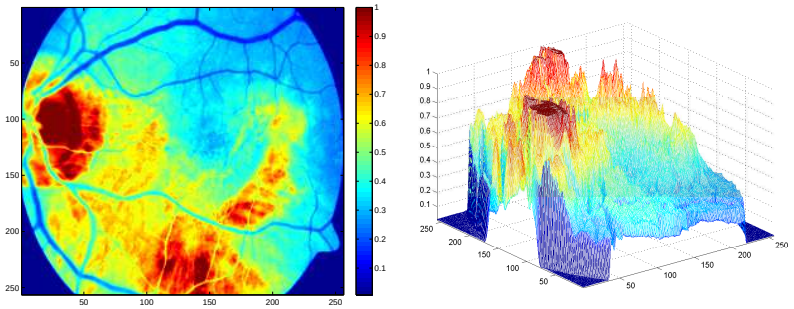
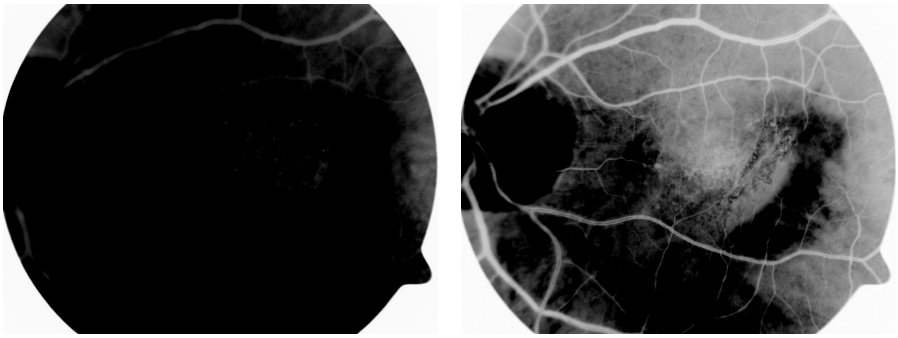
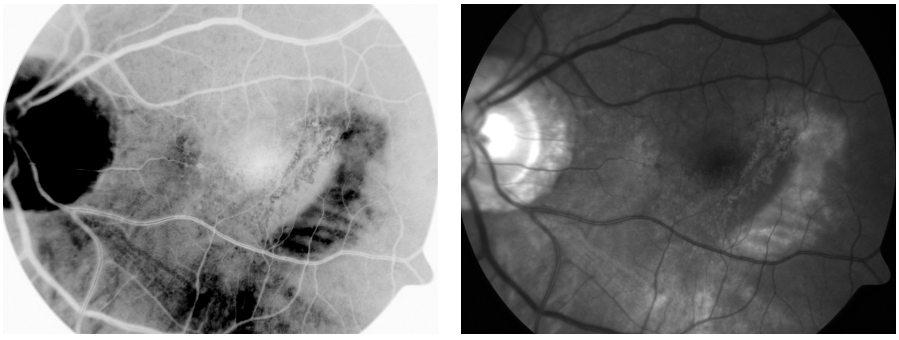


Image View Mesh View

Figure 2.3: Two different views of the surface representation of the green channel of the Colour Fundus image given in Figure 2.2c.



Cyan Channel u^C Magenta Channel u^M



Yellow Channel u^Y Black Channel u^K

Figure 2.4: Illustration of the CMYK channel representation of the Colour Fundus retina image given in Figure 2.2a.

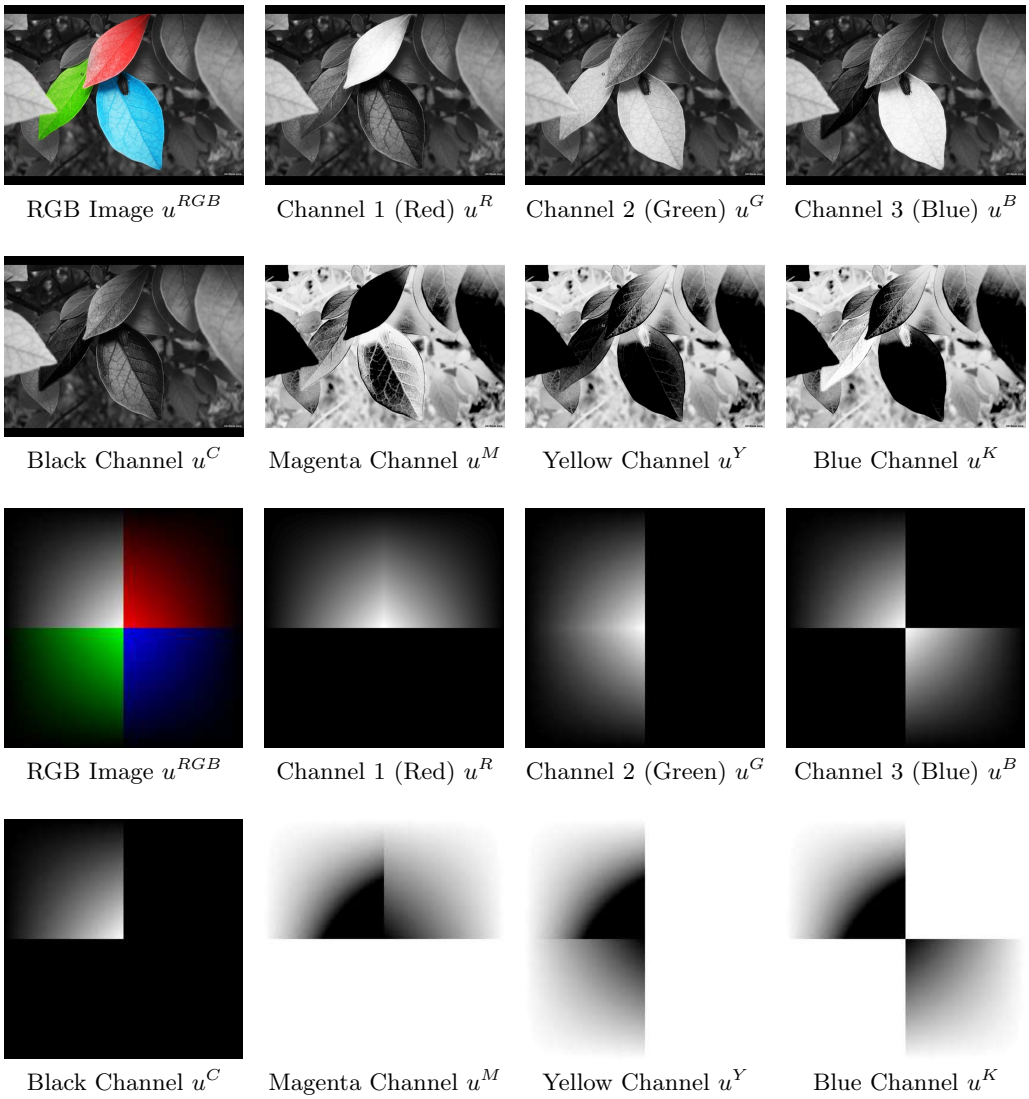
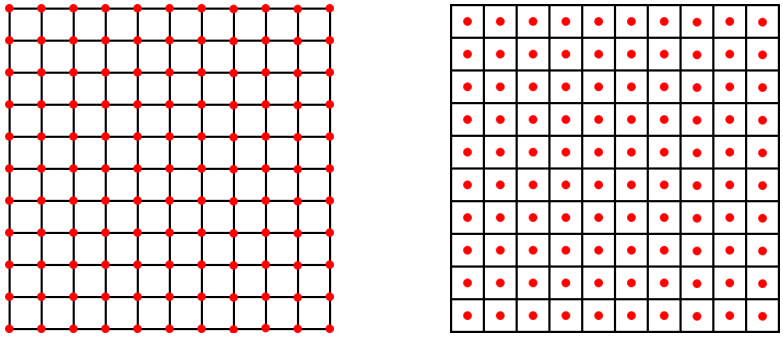


Figure 2.5: Illustration of the RGB and CYMK colour channels for two images. From top to bottom, we show 1) Colour image and RGB channels of the Leaves image; 2) CMYK channels of the Leaves image; 3) Colour image and RGB Channels of the Colourball image; 4) CMYK channels of the Colourball image.



(a) Vertex-centred (b) Cell-centred

Figure 2.6: Illustration of (a) vertex-centred discretisation and (b) cell-centred discretisation of a square mesh. Red circles show the grid points.

spacing

$$h_x = \frac{b-a}{n_x}, \qquad h_y = \frac{d-c}{n_y}$$

in the x and y-directions respectively. We may consider *vertex-centred* discretisation where points are placed on the vertices of the grid mesh giving $(n_x+1)\times(n_y+1)$ grid points (x_i, y_j) located at

$$(x_i, y_j) = (a + ih_x, c + jh_y), \qquad (i,j) \in \mathbb{B}_{n_x,n_y},$$

where $\mathbb{B}_{n_x,n_y} = \{(i,j) \in \mathbb{N}_0 \times \mathbb{N}_0 \,|\, i \le n_x,\, j \le n_y\}$. In contrast, we may also consider *cell-centred* discretisation where points are placed at the centre of the grid cells giving $n_x \times n_y$ grid points (x_i, y_j) located at

$$(x_i, y_j) = \left(a + \frac{2i-1}{2}h_x, c + \frac{2j-1}{2}h_y\right), \qquad (i,j) \in \mathbb{B}_{n_x,n_y},$$

where $\mathbb{B}_{n_x,n_y} = \{(i,j) \in \mathbb{N}_0 \times \mathbb{N}^+ \,|\, i \le n_x,\, j \le n_y\}$. See Figure 2.6 for a visual example of the discretisations.

Operators such as derivatives in the pde can then be approximated locally using the Taylor expansions

$$f(x+h,y) = \sum_{i=0}^{\infty} \frac{h^i}{i!}\frac{\partial^i f(x,y)}{\partial x^i} \qquad \text{and} \qquad f(x-h,y) = \sum_{i=0}^{\infty} (-1)^i \frac{h^i}{i!}\frac{\partial^i f(x,y)}{\partial x^i}.$$

We may then approximate the derivative $\partial f/\partial u$ at the point (i,j) as

$$\nabla_+^x (u_{i,j}) = \frac{u_{i+1,j} - u_{i,j}}{h} \qquad \text{and} \qquad \nabla_-^x (u_{i,j}) = \frac{u_{i,j} - u_{i-1,j}}{h}.$$

Building on this, we can also approximate the second derivative as

$$\Delta_c^x (u_{i,j}) = \nabla_-^x \left(\nabla_+^x (u_{i,j})\right) = \frac{u_{i+1,j} - u_{i,j} + u_{i-1,j}}{h^2}.$$

2.8 Convolution

In this section, we briefly define the convolution of two functions and introduce some relevant properties.

Definition 2.8.1 (**Convolution of two functions**). *The convolution of two functions f and g is given as*

$$[f * g](x, y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x - x', y - y') g(x', y') \, \mathrm{d}x' \, \mathrm{d}y',$$

where $*$ denotes the operation of convolution.

The convolution operation satisfies the following useful properties. For a constant $c \in \mathbb{R}$ and functions $f(\boldsymbol{x})$, $g(\boldsymbol{x})$ and $h(\boldsymbol{x})$, we have:

- Commutativity:
$$[f * g](\boldsymbol{x}) = [g * f](\boldsymbol{x}).$$
- Associativity:
$$[f * (g * h)](\boldsymbol{x}) = [(f * g) * h](\boldsymbol{x}).$$
- Associativity with scalar multiplication:
$$c[f * g](\boldsymbol{x}) = [(cf) * g](\boldsymbol{x}) = [f * (cg)](\boldsymbol{x}).$$
- Distributivity:
$$[f * (g + h)](\boldsymbol{x}) = [f * g](\boldsymbol{x}) + [f * h](\boldsymbol{x}).$$
- For calculating derivatives of a convolution of two functions with respect to one of the arguments, we have
$$\frac{\partial}{\partial x}(f * g) = \frac{\partial f}{\partial x} * g = f * \frac{\partial g}{\partial x}.$$

2.8.1 Convolution Theorem

In this section, we introduce the convolution theorem which gives us a fast method of computing the convolution of two functions using Fourier transforms.

Definition 2.8.2 (**Fourier transform**). *We define the Fourier transform $\mathcal{F}$ of a function $f(x, y)$ as*

$$\hat{f}\left(\xi_x, \xi_y\right) \;=\; \mathcal{F}\left(f(x, y)\right) \;=\; \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x, y) e^{-2\pi i(\xi_x x + \xi_y y)} \, \mathrm{d}x \, \mathrm{d}y$$

and the associated inverse Fourier transform $\mathcal{F}^{-1}$ of a function $\hat{f}\left(\xi_x, \xi_y\right)$ as

$$f(x, y) \;=\; \mathcal{F}^{-1}\left(\hat{f}\left(\xi_x, \xi_y\right)\right) \;=\; \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \hat{f}\left(\xi_x, \xi_y\right) e^{2\pi i(\xi_x x + \xi_y y)} \, \mathrm{d}\xi_x \, \mathrm{d}\xi_y.$$

Given this definition, we may give the convolution theorem which states that the Fourier transform of a convolution of two functions is equal to the product of the Fourier transforms of the functions. More formally,

Theorem 2.8.3 *Let f and g be two functions, $*$ denote the operation of convolution and $\mathcal{F}$ denote the Fourier transform operator. Then we have the relation*

$$\mathcal{F}(f * g) \;=\; \mathcal{F}(f) \cdot \mathcal{F}(g)$$

where $\cdot$ denotes point-wise multiplication.

Proof. Let $f(\boldsymbol{x})$ and $g(\boldsymbol{x})$ be two functions with associated Fourier transforms $\hat{f}(\boldsymbol{\xi})$ and $\hat{g}(\boldsymbol{\xi})$ where $\boldsymbol{x} = (x_1, x_2)$ and $\boldsymbol{\xi} = (\xi_{x_1}, \xi_{x_2})$. The the Fourier transform of the convolution of f and g is given as

$$\begin{aligned}
\mathcal{F}([f * g](\boldsymbol{x})) &= \mathcal{F}\left(\int_{-\infty}^{\infty} f(\boldsymbol{x} - \boldsymbol{x}')g(\boldsymbol{x}')\,\mathrm{d}\boldsymbol{x}'\right) \\
&= \int_{-\infty}^{\infty}\left(\int_{-\infty}^{\infty} f(\boldsymbol{x} - \boldsymbol{x}')g(\boldsymbol{x}')\,\mathrm{d}\boldsymbol{x}'\right) e^{-2\pi i(\boldsymbol{\xi}\cdot\boldsymbol{x})}\,\mathrm{d}\boldsymbol{x} \\
&= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(\boldsymbol{x} - \boldsymbol{x}')g(\boldsymbol{x}')e^{-2\pi i(\boldsymbol{\xi}\cdot\boldsymbol{x})}\,\mathrm{d}\boldsymbol{x}'\,\mathrm{d}\boldsymbol{x} \qquad (2.35)
\end{aligned}$$

where $\boldsymbol{\xi}\cdot\boldsymbol{x} = \xi_{x_1}x_1 + \xi_{x_2}x_2$. Making the substitution $\boldsymbol{y} = \boldsymbol{x} - \boldsymbol{x}'$ such that $\mathrm{d}\boldsymbol{y} = \mathrm{d}\boldsymbol{x}$, (2.35) becomes

$$\begin{aligned}
&\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(\boldsymbol{y})g(\boldsymbol{x}')e^{-2\pi i(\boldsymbol{\xi}\cdot(\boldsymbol{y}+\boldsymbol{x}'))}\,\mathrm{d}\boldsymbol{x}'\,\mathrm{d}\boldsymbol{y} \\
&= \int_{-\infty}^{\infty} f(\boldsymbol{y})e^{-2\pi i(\boldsymbol{\xi}\cdot\boldsymbol{y})}\,\mathrm{d}\boldsymbol{y}\int_{-\infty}^{\infty} g(\boldsymbol{x}')e^{-2\pi i(\boldsymbol{\xi}\cdot\boldsymbol{x}')}\,\mathrm{d}\boldsymbol{x}' \\
&= \mathcal{F}(f(\boldsymbol{y}))\,\mathcal{F}\left(g(\boldsymbol{x}')\right)
\end{aligned}$$

as required. ∎

2.9 Iterative Methods for Solving Linear Systems of Equations

In this section, we introduce some well-known iterative methods for finding solutions to linear systems of equations

$$\underbrace{\begin{pmatrix} a_{1,1} & \dots & a_{1,j} & \dots & a_{1,n} \\ \vdots & & \vdots & & \vdots \\ a_{i,1} & \dots & a_{2,j} & \dots & a_{i,n} \\ \vdots & & \vdots & & \vdots \\ a_{m,1} & \dots & a_{m,j} & \dots & a_{m,n} \end{pmatrix}\begin{pmatrix} x_1 \\ \vdots \\ x_i \\ \vdots \\ x_m \end{pmatrix} = \begin{pmatrix} b_1 \\ \vdots \\ b_i \\ \vdots \\ b_m \end{pmatrix}}_{A\boldsymbol{x}=\boldsymbol{b}}. \qquad (2.36)$$

Such strategies typically involve iterations of the form

$$\boldsymbol{x}^{(k)} = T\boldsymbol{x}^{(k-1)} + \boldsymbol{c} \tag{2.37}$$

where k is the current iterative step, T is a particular matrix and c a particular vector, both defined by the solution method and where an initial estimate $\boldsymbol{x}^{(0)}$ is chosen.

We introduce the Jacobi method which has the benefit of simple implementation, followed by an improvement of this given by the Gauss-Seidel method. We then introduce some definitions leading to the convergence properties of these methods.

2.9.1 The Jacobi Method (JAC)

The Jacobi method, named after Carl Gustav Jakob Jacobi, is an iterative method which is simple to implement and often forms the basis for other methods. It is an algorithm for determining the solutions of a system of linear equations which, when given in matrix form, is diagonally dominant. One solves for each diagonal element and substitutes an approximate value. This process is reiterated until a prescribed convergence criteria is reached.

Given a system of linear equations (2.36), we can calculate simply by matrix multiplication that the ith equation is given by

$$\sum_{j=1}^{n} a_{i,j}x_j = b_i. \tag{2.38}$$

Solving (2.38) now for x_i, we can give the equation

$$x_i = \frac{b_i}{a_{i,i}} + \sum_{\substack{j=1 \\ j\neq i}}^{n} \frac{-a_{i,j}x_j}{a_{i,i}}. \tag{2.39}$$

Given this, and assuming that we have the previous l iterations $\boldsymbol{x}^{(l)}$ $(1 \leq l \leq k-1)$, we form our iterative scheme by using previously calculated approximations of x_j on the right hand side of (2.39) which allows us to generalise the kth iterative step $x_i^{(k)}$ as follows:

$$x_i^{(k)} = \frac{1}{a_{i,i}} \left(b_i + \sum_{\substack{j=1 \\ j\neq i}}^{n} -a_{i,j}x_j^{(k-1)} \right), \qquad \text{for } i = 1, 2, \ldots, n. \tag{2.40}$$

We may now write equation (2.40) in the matrix form given by (2.37). Decomposing the matrix A into the diagonal component D and remainder R such that $A = D + R$,

$$D = \begin{pmatrix} a_{1,1} & 0 & \cdots & 0 \\ 0 & a_{2,2} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & a_{n,n} \end{pmatrix}, \qquad R = \begin{pmatrix} 0 & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & a_{n-1,n} \\ a_{n,1} & \cdots & a_{n,n-1} & 0 \end{pmatrix},$$

we may obtain the solution iteratively since

$$(D+R)\boldsymbol{x}=\boldsymbol{b} \iff D\boldsymbol{x}=\boldsymbol{b}-R\boldsymbol{x} \iff \boldsymbol{x}=D^{-1}(\boldsymbol{b}-R\boldsymbol{x}). \tag{2.41}$$

Note that we may further decompose the remainder matrix R into the sum of its strictly upper and strictly lower triangular components. Denoting the negative of these components by U and L respectively, such that $-U-L=R$, we may rewrite equation (2.41) as

$$\boldsymbol{x}=D^{-1}(L+U)\boldsymbol{x}+D^{-1}\boldsymbol{b}. \tag{2.42}$$

In the form of equation 2.37, the Jacobi method may thus be given as

$$\boldsymbol{x}=T_J\boldsymbol{x}+\boldsymbol{c}_J \qquad \text{where} \qquad T_J=D^{-1}(L+U) \qquad \text{and} \qquad \boldsymbol{c}_J=D^{-1}\boldsymbol{b}. \tag{2.43}$$

For numerical implementation, this method is typically implemented in practice using equation (2.40) since a significant amount of memory may be required to form the matrices of equation (2.43). This practice also allows the implementation of parallel computing thereby potentially increasing the speed at which the method may be implemented. Additionally, we may note that the computation of $x_i^{(k+1)}$ requires the values of each entry of $\boldsymbol{x}^{(k)}$ except for $x_i^{(k)}$ whereas the Gauss-Seidel method, which will be considered below, requires the entry $x_i^{(k)}$ to calculate $x_i^{(k+1)}$. The algorithm for the Jacobi method is presented in Algorithm 1.

Algorithm 1 Jacobi Method (JAC)

function $\boldsymbol{x} \leftarrow$JAC$(\boldsymbol{x}^{(0)}, A, \boldsymbol{b}, maxit, tol)$
  for $k \leftarrow 1 : maxit$ **do**
    Compute $\boldsymbol{x}^{(k)}$ given $\boldsymbol{x}^{(k-1)}$ using equation (2.40):

$$x_i^{(k)} \leftarrow \frac{1}{a_{i,i}}\left(b_i+\sum_{\substack{j=1\\ j\neq i}}^{n} -a_{i,j}x_j^{(k-1)}\right), \qquad \text{for } i=1,2,\ldots,n$$

    Check convergence:
    if $\|\boldsymbol{x}^{(k)}-\boldsymbol{x}^{(k-1)}\| < tol$ or $k > maxit$ **then**
      break
    end if
  end for
  $\boldsymbol{x} \leftarrow \boldsymbol{x}^{(k)}$
end function

2.9.2 The Gauss-Seidel Method (GS)

The Gauss-Seidel method, named after Carl Friedrich Gauss and Philipp Ludwig von Seidel, and is similar to the Jacobi method and in fact improves on it. Unlike the Jacobi method, Gauss-Seidel uses the values of the current step where possible rather than

the previous step, i.e. to calculate $x_i^{(k)}$, Gauss-Seidel uses $x_1^{(k)}, \ldots, x_{i-1}^{(k)}$ while Jacobi uses $x_1^{(k-1)}, \ldots, x_{i-1}^{(k-1)}$ which should be a less accurate approximation to $x_1, \ldots, x_{i-1}$. Gauss-Seidel can be written similarly to the Jacobi method as

$$x_i^{(k)} = \frac{1}{a_{i,i}} \left(b_i + \sum_{j=1}^{i-1} -a_{i,j} x_j^{(k)} + \sum_{j=i+1}^{n} -a_{i,j} x_j^{(k-1)} \right), \qquad \text{for } i = 1, 2, \ldots, n. \quad (2.44)$$

It is clear that with Gauss-Seidel each entry $x_i^{(k)}$ is strongly dependent on all values $x_j^{(k)}$ $(1 \leq j \leq i-1)$ which must already be calculated. The order in which the equations are evaluated is therefore extremely important. In order to obtain the matrix form of this method, we multiply equation (2.44) by the denominator $a_{i,i}$ and move the components from the current iteration to the left hand side, obtaining

$$a_{i,i} x_i^{(k)} - \sum_{j=1}^{i-1} -a_{i,j} x_j^{(k)} = b_i + \sum_{j=i+1}^{n} -a_{i,j} x_j^{(k-1)}, \qquad \text{for } i = 1, 2, \ldots, n, \quad (2.45)$$

which can clearly be written as

$$(D - L)\boldsymbol{x}^{(k)} = U\boldsymbol{x}^{(k-1)} + \boldsymbol{b}$$

or, in the form of equation 2.37, the Gauss-Seidel method may be given as

$$\boldsymbol{x}^{(k)} = T_{GS}\boldsymbol{x}^{(k-1)} + \boldsymbol{c}_{GS} \qquad \text{where } T_{GS} = (D - L)^{-1}(U), \boldsymbol{c}_{GS} = (D - L)^{-1}\boldsymbol{b}. \quad (2.46)$$

As with the Jacobi method, Gauss-Seidel is typically implemented in practice using equation (2.44) in order to reduce the amount of memory required. Additionally, there is no need to allocate memory to two different arrays $x^{(k)}$ and $x^{(k-1)}$ since we may simply successively overwrite $x^{(k-1)}$ with the newly computed entries for $x^{(k)}$. This can be particularly useful for solving large systems of equations. The algorithm for the Gauss-Seidel method is presented in Algorithm 2.

2.9.3 Lexicographic Ordering

Lexicographic ordering (or dictionary ordering), defined below, describes the order of the grid points as increasing from left to right and from bottom to top. In Gauss-Seidel, if this ordering system is followed then equation (2.44) would first update the bottom-left point before moving on to the point to the right, and finishing at the top-right point.

Definition 2.9.1 *An ordering for the Cartesian product $\times$ of any two sets A and B with order relations $< A$ and $< B$ respectively such that if $(a_1, b_1) \in A \times B$ and $(a_2, b_2) \in A \times B$, then $(a_1, b_1) < (a_2, b_2)$ if and only if either*

- $a_1 < A a_2$ or
- $a_1 = a_2$ *and* $b_1 < B b_2$.

Algorithm 2 Gauss-Seidel Method (GS)

function $\boldsymbol{x} \leftarrow \text{GS}(\boldsymbol{x}^{(0)}, A, \boldsymbol{b}, maxit, tol)$
  for $k \leftarrow 1 : maxit$ **do**
    Compute $\boldsymbol{x}^{(k)}$ given $\boldsymbol{x}^{(k-1)}$ and previously calculated components of $\boldsymbol{x}^{(k)}$ using equation (2.44):

$$x_i^{(k)} \leftarrow \frac{1}{a_{i,i}} \left(b_i + \sum_{j=1}^{i-1} -a_{i,j} x_j^{(k)} + \sum_{j=i+1}^{n} -a_{i,j} x_j^{(k-1)} \right), \qquad \text{for } i = 1, 2, \ldots, n$$

    Check convergence:
    if $\|\boldsymbol{b} - \boldsymbol{x}^{(k)}\| < tol$ or $k > maxit$ **then**
      break
    end if
  end for
  $\boldsymbol{x} \leftarrow \boldsymbol{x}^{(k)}$
end function

In practice, typically either this system is used or a system known as red-black ordering is used. Red-black ordering involves viewing the points as a checkboard made up of alternate red and black points. All of the red points are updated first and then the black points are updated. This system allows for Gauss-Seidel to run in parallel and is demonstrated in Figure 2.7 below.

2.9.4 Convergence

Each of the iterative methods given above define iterative sequences of the form

$$\boldsymbol{x}^{(k)} = T\boldsymbol{x}^{(k-1)} + \boldsymbol{c}$$

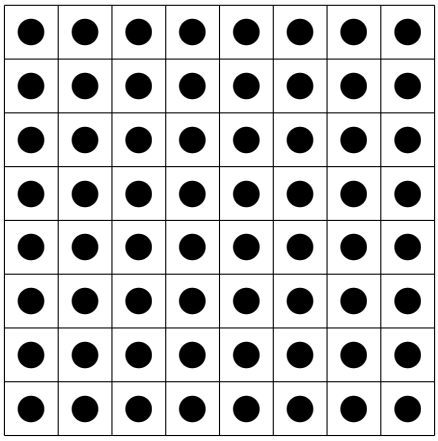
where T is a particular iteration matrix. In this section, we consider the conditions which must be satisfied in order for the method to converge, i.e. under what conditions do we have $\lim_{k\to\infty} \boldsymbol{x}^k = \bar{\boldsymbol{x}}$ where $\bar{\boldsymbol{x}}$ denotes the true solution. We first consider some relevant properties of matrices.

Definition 2.9.2 (Symmetric Matrix) *Let A be a square matrix. A is called* symmetric *if it is equal to its transpose, i.e. $A = A^T$.*

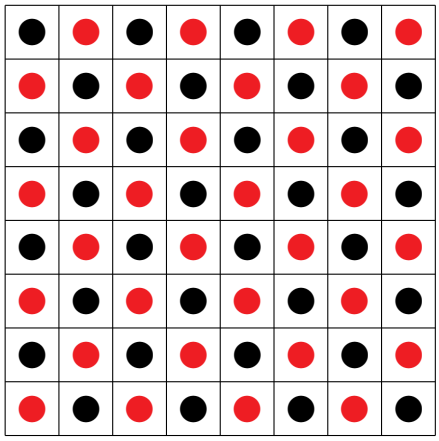
Definition 2.9.3 (Diagonally Dominant Matrix) *A matrix A is called* diagonally dominant *if for each row the absolute value of the entry on the diagonal is greater than or equal to the sum of the absolute values of the the off-diagonal entries, i.e.*

$$|a_{i,i}| \geq \sum_{i \neq j} |a_{i,j}| \ \forall i.$$

A is called strictly diagonally dominant *if for each row the absolute value of the entry on the diagonal is greater than the sum of the absolute values of the the off-diagonal*



Lexicographic Ordering



Red-Black Ordering

57	58	59	60	61	62	63	64
49	50	51	52	53	54	55	56
41	42	43	44	45	46	47	48
33	34	35	36	37	38	39	40
25	26	27	28	29	30	31	32
17	18	19	20	21	22	23	24
9	10	11	12	13	14	15	16
1	2	3	4	5	6	7	8

Lexicographic Ordering

61	29	62	30	63	31	64	32
25	57	26	58	27	59	28	60
53	21	54	22	55	23	56	24
17	49	18	50	19	51	20	52
45	13	46	14	47	15	48	16
9	41	10	42	11	43	12	44
37	5	38	6	39	7	40	8
1	33	2	34	3	35	4	36

Red-Black Ordering

Figure 2.7: Lexicographic ordering and Red-Black ordering for an 8×8 example. On both rows, the figures on the left show Lexicographic ordering and the figures on the right show Red-Black ordering.

entries, i.e.

$$|a_{i,i}| > \sum_{i \neq j} |a_{i,j}| \ \forall i.$$

Remark 2.9.4 *Let A be a* strictly *diagonally dominant matrix. Then*

1. *A is non-singular and so the system $A\boldsymbol{x} = \boldsymbol{b}$ has a unique solution.*
2. *The Jacobi and Gauss-Seidel methods converge for the linear system $A\boldsymbol{x} = \boldsymbol{b}$.*

Definition 2.9.5 (Eigenvalue, Eigenvector) *Let A be a matrix and $\boldsymbol{x} \in \mathbb{R}^n$ a vector such that $A\boldsymbol{x} = \lambda\boldsymbol{x}$. Then λ is called an eigenvalue of A and $\boldsymbol{x}$ the corresponding eigenvector.*

Definition 2.9.6 (Positive Definite Matrix) *Let A be a real symmetric matrix. A is called* positive definite *if for all non-zero vectors $\boldsymbol{v}$, we have $\boldsymbol{v}^T A\boldsymbol{v} > 0$. This is equivalent to saying that all of the eigenvalues of the matrix A are positive.*

Remark 2.9.7 *Let A be a Hermitian positive definite matrix. Then A has the following properties:*

1. *All of the eigenvalues of the matrix A are positive.*
2. *A has a unique Cholesky decomposition.*
3. *The inverse A^{-1} of the matrix A exists.*

We now introduce the main result which is a necessary and sufficient condition for the convergence of iterative methods.

Lemma 2.9.8 *Let A be a matrix with eigenvalues λ_i. Then*

$$\|A\| < 1 \iff \rho(A) = \max_i |\lambda_i| < 1.$$

$\rho(A)$ is called the spectral radius of A and $\|\cdot\|$ denotes an operator norm, i.e. $\|A\| = \max_{\boldsymbol{v} \neq 0} \frac{\|A\boldsymbol{v}\|}{\|\boldsymbol{v}\|}$.

Theorem 2.9.9 *Let $A \in \mathbb{R}^n$ be a matrix. Then*

$$\lim_{k \to \infty} A^k = 0 \iff \rho(A) < 1.$$

Theorem 2.9.10 (Spectral Radius and Iterative Convergence) *For any $\boldsymbol{x}^{(0)} \in \mathbb{R}^n$, the sequence $\left\{\boldsymbol{x}^{(k)}\right\}_{k=0}^{\infty}$ defined by*

$$\boldsymbol{x}^{(k)} = T\boldsymbol{x}^{(k-1)} + \boldsymbol{c}, \qquad k \geq 1 \tag{2.47}$$

converges to the unique solution $\boldsymbol{x}^* = T\boldsymbol{x}^* + \boldsymbol{c}$ if and only if $\rho(T) < 1$.

Proof. Let $\left\{\boldsymbol{x}^{(k)}\right\}_{k=0}^{\infty}$ be a sequence defined by (2.47) and let x^* denote the true solution which satisfies $\boldsymbol{x}^* = T\boldsymbol{x}^* + \boldsymbol{c}$. We may calculate the error vector $\boldsymbol{e}$ at iteration k as

$$\overbrace{x^* - x^{(k)}}^{e^{(k)}} = \overbrace{T(x^* - x^{(k-1)})}^{Te^{(k-1)}} = \ldots = \overbrace{T^k(x^* - x^{(0)})}^{T^k e^{(0)}}.$$

In the forward direction, we assume that the sequence converges and we wish to show that the spectral radius is less than 1. Indeed, since the convergence of the sequence implies that the error value tends to zero while $e^{(0)}$ is a fixed, non-zero vector it follows that the limit of T^k as k tends to infinity is the zero matrix. By Theorem 2.9.9, the spectral radius is less than 1. In the other direction, we assume that the spectral radius is less than 1 which, by Theorem 2.9.9, means that the limit T^k as k tends to infinity must be 0. Then we have

$$\left\|e^k\right\| = \left\|T^k e^{(0)}\right\| \leq \left\|T^k\right\| \left\|e^{(0)}\right\|$$

and so the norm of the error is bounded by zero. ∎

Theorem 2.9.11 (Convergence of Gauss-Seidel) *Given a system of equations $A\boldsymbol{x} = \boldsymbol{b}$ (2.36), if the matrix A is strictly diagonally dominant then Gauss-Seidel will converge for any initial estimate $\boldsymbol{x}^{(0)}$.*

Proof. Let A be strictly diagonally dominant, that is

$$|a_{i,i}| > \sum_{\substack{j=1 \\ j \neq i}}^{n} |a_{i,j}|, \qquad i = 1, \ldots, n,$$

and recall that for Gauss-Seidel, our iteration step is given by $\boldsymbol{x}^{(k)} = T_{GS}\boldsymbol{x}^{(k-1)} + c_{GS}$ where $T_{GS} = (D - L)^{-1}U$ and L, D, U are the lower triangular, diagonal and upper triangular components of A respectively. We would like to show that Gauss-Seidel will converge for any initial estimate $\boldsymbol{x}^{(0)}$. We proceed by showing that the maximum eigenvalue, and hence the spectral radius, of T_{GS} is less than 1. Indeed, letting λ and

$\boldsymbol{x}$ denote the eigenvalues and corresponding eigenvectors of T_{GS} respectively, we have

$$\begin{aligned}
T_{GS}\boldsymbol{x} = \lambda\boldsymbol{x} \Leftrightarrow\ & (D-L)^{-1}U\boldsymbol{x} = \lambda\boldsymbol{x} \ \Leftrightarrow\ U\boldsymbol{x} = (D-L)\lambda\boldsymbol{x} \\
\Leftrightarrow\ & \sum_{j>\ell} a_{\ell,j}x_j = \lambda\left(a_{\ell,\ell}x_\ell - \sum_{j<\ell} a_{\ell,j}x_j\right) \ \forall\ell \\
\Leftrightarrow\ & \left|\sum_{j>\ell} a_{\ell,j}x_j\right| = |\lambda|\left|\left(a_{\ell,\ell}x_\ell - \sum_{j<\ell} a_{\ell,j}x_j\right)\right| \\
\Leftrightarrow\ & \sum_{j>\ell} |a_{\ell,j}x_j| \geq |\lambda|\left(|a_{\ell,\ell}x_\ell| - \left|\sum_{j<\ell} a_{\ell,j}x_j\right|\right) \\
\Leftrightarrow\ & \sum_{j>\ell} |a_{\ell,j}|\,|x_j| \geq |\lambda|\left(|a_{\ell,\ell}|\,|x_\ell| - \sum_{j<\ell} |a_{\ell,j}|\,|x_j|\right) \\
\Leftrightarrow\ & \sum_{j>\ell} |a_{\ell,j}| \geq |\lambda|\left(|a_{\ell,\ell}| - \sum_{j<\ell} |a_{\ell,j}|\right) \qquad (2.48)
\end{aligned}$$

since $|x_n| = 1$ and $|x_j| \leq 1$. Since A is assumed to be diagonally dominant, we have

$$|a_{\ell,\ell}| > \sum_{j>\ell} |a_{\ell,j}| + \sum_{j<\ell} |a_{\ell,j}| \ \Leftrightarrow\ |a_{\ell,\ell}| - \sum_{j<\ell} |a_{\ell,j}| > \sum_{j>\ell} |a_{\ell,j}|.$$

Therefore, in order for (2.48) to be satisfied, we must have $|\lambda| < 1$ and hence the spectral radius of T_{GS} must be less than 1. By Theorem 2.9.10, the method converges. ∎

Remark 2.9.12

- *We can give a convergence proof of the Jacobi method similarly.*
- *If the matrix A of the above system is strictly diagonally dominant, both the Gauss-Seidel and Jacobi methods converge. Since the spectral radius of the iteration matrices for Gauss-Seidel is lower than that of Jacobi, Gauss-Seidel will achieve faster convergence.*

2.10 Iterative Solutions of Nonlinear Equations

In this section, we consider the issue of finding solutions to systems of non-linear equations. Let $F : \Omega \subset \mathbb{R}^n \to \mathbb{R}^n$ be a non-linear operator which is continuously differentiable on $\mathbb{R}^n$. We represent a non-linear system of equations as $F(\boldsymbol{x}) = \boldsymbol{y}$, where $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$ are vectors with $\boldsymbol{x}$ typically unknown and $\boldsymbol{y}$ a fixed known vector. Since we may simply incorporate the fixed term $\boldsymbol{y}$ into F, we can write this equation in the form

$$F(\boldsymbol{x}) = \mathbf{0} \qquad (2.49)$$

without loss of generality where $\mathbf{0} \in \mathbb{R}^n$ is the vector of zeros. Our aim is to find the solution $x^* \in \mathbb{R}^n$ of Equation (2.49). We begin by presenting Newton's method which aims to solve a system of non-linear equations iteratively by finding closer approximations to the solution. We will then proceed with the descent method with Gradient Descent and Time Marching as two particular cases of this. We then introduce the conjugate gradient method for solving systems with symmetric positive definite matrices before mentioning additive operator splitting (AOS).

2.10.1 Newton's Method

Let F be a non-linear operator as shown above and let $J = (\partial F_i/\partial x_j)$ denote its Jacobian matrix. Assume also that J is Lipschitz continuous, i.e. there exists $\beta \geq 0$ such that

$$||J(\boldsymbol{x}) - J(\boldsymbol{y})|| \leq \beta ||\boldsymbol{x} - \boldsymbol{y}|| \qquad \forall\, \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$$

for any operator norm. Newton's method aims to solve the non-linear system $F(\boldsymbol{x}) = \mathbf{0}$ by beginning with an initial approximation $\boldsymbol{x}^{(0)}$ of the solution and carrying out iterations in the following manner:

$$\boldsymbol{x}^{(k)} = \boldsymbol{x}^{(k-1)} - J\left(\boldsymbol{x}^{(k-1)}\right)^{-1} F\left(\boldsymbol{x}^{(k-1)}\right), \qquad k = 1, 2, \ldots. \tag{2.50}$$

The aim here is to find successively closer approximations to the solution $\boldsymbol{x}^*$. Computing the inverse of the Jacobian can however be a difficult task which we would like to avoid. We can achieve this by rewriting (2.50) as

$$J\left(\boldsymbol{x}^{(k-1)}\right)\left(\boldsymbol{x}^{(k)} - \boldsymbol{x}^{(k-1)}\right) = -F\left(\boldsymbol{x}^{(k-1)}\right). \tag{2.51}$$

We then simply need to calculate the Jacobian at the previous step, as shown in Algorithm 3.

Algorithm 3 Newton's Method (NEWT)

function $\boldsymbol{x} \leftarrow$NEWT$(\boldsymbol{x}^{(0)}, maxit, tol)$
  for $k \leftarrow 1 : maxit$ **do**
    Solve $J(\boldsymbol{x}^{(k-1)})\boldsymbol{v}^{(k)} = -F(\boldsymbol{x}^{(k-1)})$ for $\boldsymbol{v}^{(k)}$
    Update $\boldsymbol{x}^{(k)} = \boldsymbol{x}^{(k-1)} + \boldsymbol{v}^{(k)}$
    Check convergence:
    if $\|\boldsymbol{x}^{(k)} - \boldsymbol{x}^{(k-1)\|} < tol$ or $k > maxit$ **then**
      break
    end if
  end for
  $\boldsymbol{x} \leftarrow \boldsymbol{x}^{(k)}$
end function

Assuming that the initial estimate is sufficiently close to the true solution, Newton's method can offer fast convergence [99]. Moreover, if the Jacobian is non-singular at

the solution, local quadratic convergence can be proved [99, 150]. However, since the Jacobian must be calculated at each step, this is computationally expensive and can prove to be ill-conditioned.

2.10.2 Steepest Descent Method

Similar to Newton's method, the descent method requires an initial estimate $\boldsymbol{x}^{(0)} \in \mathbb{R}^n$ of the solution and then proceeds iteratively to approximate the solution. In this case the iteration scheme takes the form

$$\boldsymbol{x}^{(k)} = \boldsymbol{x}^{(k-1)} - \alpha^{(k-1)}\boldsymbol{s}^{(k-1)}, \qquad k = 1, 2, 3, \ldots$$

where $\alpha^{(k-1)} \in \mathbb{R}_{>0}$, known as the *step length*, is a positive scalar which is not fixed and may take different values at each iteration, and $-\boldsymbol{s}^{(k-1)}$ is a pre-defined *search direction* which we use to find the new iterate $\boldsymbol{x}^{(k)}$.

A particular choice of search direction yields the *Steepest Descent* (or *Gradient Descent*) *Method*. For this method, the search direction is given as the opposite direction to the gradient $\nabla F\left(\boldsymbol{x}^{(k-1)}\right)$ since F should decrease faster along this.

We therefore present the iterative scheme of the steepest descent method as

$$\boldsymbol{x}^{(k)} = \boldsymbol{x}^{(k-1)} - \alpha^{(k-1)}\nabla F(\boldsymbol{x}^{(k-1)}), \qquad k = 1, 2, 3, \ldots \tag{2.52}$$

For descent methods, we expect that the function value is decreased with each iteration, i.e. $F(\boldsymbol{x}^{(k)}) \leq F(\boldsymbol{x}^{(k-1)})$, and choose the step length $\alpha^{(k-1)}$ to be sufficiently small to satisfy this.

If we fix the step length to be equal to some time step Δt, then we obtain the *Time Marching* method. We begin at time $t = 0$ and proceed in time until a solution is obtained. One limitation of this model is that the time step must be chosen sufficiently small for the model to remain stable at each iteration which can increase the number of steps required to achieve a solution dramatically.

2.10.3 Conjugate Gradient Method

In this section we introduce the Conjugate Gradient method developed by Hestenes and Stiefel [98]. This method offers a fast solution strategy for non-linear problems whose matrix is symmetric positive definite (spd). Suppose that we want to solve the linear system

$$A\boldsymbol{x} = \boldsymbol{b} \tag{2.53}$$

where A is an $n \times n$ symmetric positive definite matrix, $\boldsymbol{b}$ is a known vector of n elements and $\boldsymbol{x}$ is an unknown vector of n elements which we wish to find. Solving this linear system is equivalent to finding the minimum of the quadratic function

$$f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^T A\boldsymbol{x} - \boldsymbol{x}^T b. \tag{2.54}$$

That is, we find the stationary points of $f(\boldsymbol{x})$ in (2.54) which occur at $f'(\boldsymbol{x}) = A\boldsymbol{x} - \boldsymbol{b} = \boldsymbol{0}$. Note that $f(\boldsymbol{x})$ has only one stationary point since it is quadratic. Additionally, we know that this stationary point is a minimum because A is positive definite. A line search optimisation method can therefore be used to minimise $f(\boldsymbol{x})$, that is starting with an initial estimate, $\boldsymbol{x}$ is updated as follows:

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \alpha_k \boldsymbol{p}_k$$

where $\boldsymbol{p}_k$ is a search direction and α is a line search parameter chosen to minimise $f(\boldsymbol{x}_{k+1}) = f(\boldsymbol{x}_k + \alpha_k \boldsymbol{p}_k)$ along the direction $\boldsymbol{p}_k$.

Given $f(\boldsymbol{x})$, we can chose α analytically so there is no need to perform such a line search. The minimum of $f(\boldsymbol{x}_{k+1})$ over α occurs when the residual

$$\boldsymbol{r}_{k+1} = r(\boldsymbol{x}_{k+1}) = \boldsymbol{b} - A\boldsymbol{x}_{k+1} = -\nabla f(\boldsymbol{x}_{k+1})$$

is orthogonal to the search direction $\boldsymbol{p}_k$. That is

$$\frac{d}{d\alpha} f(\boldsymbol{x}_{k+1}) = \nabla f(\boldsymbol{x}_{k+1}) \frac{d}{d\alpha_k} \boldsymbol{x}_{k+1} = -\boldsymbol{r}_{k+1}^T \frac{d}{d\alpha_k} (\boldsymbol{x}_k + \alpha_k \boldsymbol{p}_k) = -\boldsymbol{r}_{k+1}^T \boldsymbol{p}_k = 0$$

where we can write the $(k+1)$th iterate of the residual as

$$\boldsymbol{r}_{k+1} = \boldsymbol{b} - A\boldsymbol{x}_{k+1} = \boldsymbol{b} - A(\boldsymbol{x}_k + \alpha_k \boldsymbol{p}_k) = \boldsymbol{r}_k - \alpha_k A \boldsymbol{p}_k. \tag{2.55}$$

The minimum value of α_k is then given when

$$(\boldsymbol{r}_k - \alpha_k A\boldsymbol{p}_k)^T \boldsymbol{p}_k = 0 \Leftrightarrow \left(\boldsymbol{r}_k^T - \alpha_k \boldsymbol{p}_k^T A\right) \boldsymbol{p}_k = 0 \Leftrightarrow \alpha_k \boldsymbol{p}_k^T A \boldsymbol{p}_k = \boldsymbol{r}_k^T \boldsymbol{p}_k \Leftrightarrow \alpha_k = \frac{\boldsymbol{r}_k^T \boldsymbol{p}_k}{\boldsymbol{p}_k^T A \boldsymbol{p}_k}.$$

Now that we have an expression for α_k, we address the question of updating the directions $\boldsymbol{p}_k$. We require each direction to be conjugate to all previous directions which can be given by

$$\boldsymbol{p}_k = \boldsymbol{r}_k - \sum_{i<k} \frac{\boldsymbol{p}_i^T A \boldsymbol{r}_k}{\boldsymbol{p}_i^T A \boldsymbol{p}_i} \boldsymbol{p}_i. \tag{2.56}$$

However, it is not convenient to store all previous directions so we aim to avoid this by writing

$$\boldsymbol{p}_{k+1} = \boldsymbol{r}_{k+1} + \beta_k \boldsymbol{p}_k \tag{2.57}$$

for some β_k. The Fletcher-Reeves formula for the β_k term is given as

$$\beta_k = \frac{\boldsymbol{r}_{k+1}^T \boldsymbol{r}_{k+1}}{\boldsymbol{r}_k^T \boldsymbol{r}_k}. \tag{2.58}$$

We can show that this is correct by induction as follows:

Proof. **Case $k = 1$.** In order to determine β_1, we can calculate the first and second

iterates of $\boldsymbol{p}_k$ using (2.56) as

$$\boldsymbol{p}_1 = \boldsymbol{r}_1 - \sum_{i<1} \frac{\boldsymbol{p}_i^T A \boldsymbol{r}_1}{\boldsymbol{p}_i^T A \boldsymbol{p}_i} \boldsymbol{p}_i = \boldsymbol{r}_1. \tag{2.59}$$

$$\boldsymbol{p}_2 = \boldsymbol{r}_2 - \sum_{i<2} \frac{\boldsymbol{p}_i^T A \boldsymbol{r}_2}{\boldsymbol{p}_i^T A \boldsymbol{p}_i} \boldsymbol{p}_i = \boldsymbol{r}_2 - \frac{\boldsymbol{p}_1^T A \boldsymbol{r}_2}{\boldsymbol{p}_1^T A \boldsymbol{p}_1} \boldsymbol{p}_1. \tag{2.60}$$

From the expression for the $(k+1)$th iterate of the residual (2.55) and the expression for the search direction (2.57), we have

$$\boldsymbol{p}_k = \frac{\boldsymbol{r}_k - \boldsymbol{r}_{k+1}}{\alpha_k A} \qquad \text{and} \qquad \boldsymbol{p}_k = \boldsymbol{r}_k + \beta_{k-1} \boldsymbol{p}_{k-1}$$

respectively and substituting these into (2.60), we can write

$$\boldsymbol{p}_2 = \boldsymbol{r}_2 - \frac{(\boldsymbol{r}_1 - \boldsymbol{r}_2)^T \boldsymbol{r}_2}{(\boldsymbol{r}_1 + \beta_0 \boldsymbol{p}_0)^T (\boldsymbol{r}_1 - \boldsymbol{r}_2)} \boldsymbol{p}_1 = \boldsymbol{r}_2 + \frac{\boldsymbol{r}_2^T \boldsymbol{r}_2}{\boldsymbol{r}_1^T \boldsymbol{r}_1} \boldsymbol{p}_1 = \boldsymbol{r}_2 + \beta_1 \boldsymbol{p}_1$$

where the β term is given by

$$\beta_1 = \frac{\boldsymbol{r}_2^T \boldsymbol{r}_2}{\boldsymbol{r}_1^T \boldsymbol{r}_1}.$$

Case $k+1$ given $j < k+1$. Assuming now that this holds for all $j < k+1$, we show that it will hold for $k+1$ as follows. We can calculate the kth and $k+1$th iterates of $\boldsymbol{p}_k$ using (2.56) as

$$\boldsymbol{p}_k = \boldsymbol{r}_k - \sum_{i<k} \frac{\boldsymbol{p}_i^T A \boldsymbol{r}_k}{\boldsymbol{p}_i^T A \boldsymbol{p}_i} \boldsymbol{p}_i.$$

$$\begin{aligned} \boldsymbol{p}_{k+1} &= \boldsymbol{r}_{k+1} - \sum_{i<k+1} \frac{\boldsymbol{p}_i^T A \boldsymbol{r}_{k+1}}{\boldsymbol{p}_i^T A \boldsymbol{p}_i} \boldsymbol{p}_i \\ &= \boldsymbol{r}_{k+1} - \frac{\boldsymbol{p}_k^T A \boldsymbol{r}_{k+1}}{\boldsymbol{p}_k^T A \boldsymbol{p}_k} \boldsymbol{p}_k - \frac{\boldsymbol{p}_{k-1}^T A \boldsymbol{r}_{k+1}}{\boldsymbol{p}_{k-1}^T A \boldsymbol{p}_{k-1}} \boldsymbol{p}_{k-1} - \ldots - \frac{\boldsymbol{p}_1^T A \boldsymbol{r}_{k+1}}{\boldsymbol{p}_1^T A \boldsymbol{p}_1} \boldsymbol{p}_1. \end{aligned} \tag{2.61}$$

From (2.55) and (2.57), we have

$$\boldsymbol{p}_k = \frac{\boldsymbol{r}_k - \boldsymbol{r}_{k+1}}{\alpha_k A} \qquad \text{and} \qquad \boldsymbol{p}_k = \boldsymbol{r}_k + \beta_{k-1} \boldsymbol{p}_{k-1}$$

respectively and substituting these into (2.61), we can write

$$
\begin{aligned}
\boldsymbol{p}_{k+1} &= \boldsymbol{r}_{k+1} - \sum_{i<k+1} \frac{\boldsymbol{p}_i^T A \boldsymbol{r}_{k+1}}{\boldsymbol{p}_i^T A \boldsymbol{p}_i} \boldsymbol{p}_i \\
&= \boldsymbol{r}_{k+1} - \sum_{i<k+1} \frac{\left(\boldsymbol{r}_i - \boldsymbol{r}_{i+1}\right)^T \boldsymbol{r}_{k+1}}{\left(\boldsymbol{r}_i + \beta_{i-1}\boldsymbol{p}_{i-1}\right)^T \left(\boldsymbol{r}_i - \boldsymbol{r}_{i+1}\right)} \boldsymbol{p}_i \\
&= \boldsymbol{r}_{k+1} - \sum_{i<k+1} \frac{-\boldsymbol{r}_{k+1}^T \boldsymbol{r}_{k+1}}{\boldsymbol{r}_i^T \boldsymbol{r}_i - \boldsymbol{r}_i^T \boldsymbol{r}_{i+1} + \beta_{i-1}\boldsymbol{p}_{i-1}^T \left(\boldsymbol{r}_i - \boldsymbol{r}_{i+1}\right)} \boldsymbol{p}_i \qquad (2.62) \\
&= \boldsymbol{r}_{k+1} + \boldsymbol{r}_{k+1}^T \boldsymbol{r}_{k+1} \sum_{i<k+1} \frac{\boldsymbol{p}_i}{\boldsymbol{r}_i^T \boldsymbol{r}_i + \beta_{i-1}\boldsymbol{p}_{i-1}^T \left(\boldsymbol{r}_i - \boldsymbol{r}_{i+1}\right)} \qquad (2.63) \\
&= \boldsymbol{r}_{k+1} + \frac{\boldsymbol{r}_{k+1}^T \boldsymbol{r}_{k+1}}{\boldsymbol{r}_k^T \boldsymbol{r}_k} \boldsymbol{p}_k \qquad (2.64)
\end{aligned}
$$

where (2.62) and (2.63) hold because $\boldsymbol{r}_{i+1}^T \boldsymbol{r}_{k+1} = 0$ for all $i < k$. To see that (2.64) holds, we would like to show that $\beta_{i-1}\boldsymbol{p}_{i-1}^T \left(\boldsymbol{r}_i - \boldsymbol{r}_{i+1}\right) = 0$ for $i = k$. Indeed, we have

$$
\begin{aligned}
\beta_{k-1}\boldsymbol{p}_{k-1}^T &= \beta_{k-1}(\boldsymbol{r}_{k-1} + \beta_{k-2}\boldsymbol{p}_{k-2})^T \\
&= \beta_{k-1}(\boldsymbol{r}_{k-1}^T + \beta_{k-2}\boldsymbol{p}_{k-2}^T) \\
&= \beta_{k-1}(\boldsymbol{r}_{k-1}^T + \beta_{k-2}(\boldsymbol{r}_{k-2}^T + \beta_{k-3}\boldsymbol{p}_{k-3}^T)) \\
&= \sum_{j=1}^{k-1} \left(\prod_{i=j}^{k-1} \beta_i \right) \boldsymbol{r}_j^T \qquad (2.65)
\end{aligned}
$$

and (2.65) holds by expansion and by (2.59). Consequently, we have

$$
\beta_{k-1}\boldsymbol{p}_{k-1}^T \left(\boldsymbol{r}_k - \boldsymbol{r}_{k+1}\right) = \left(\sum_{j=1}^{k-1} \left(\prod_{i=j}^{k-1} \beta_i \right) \boldsymbol{r}_j^T \right) \left(\boldsymbol{r}_k - \boldsymbol{r}_{k+1}\right) = 0 \qquad (2.66)
$$

since the residuals are orthogonal to the earlier ones. ∎

In order to solve (2.53), we begin in the direction given by the residual and compute the step length α_k and update $\boldsymbol{x}_{k+1}$ and the residual. Then compute the term β_k and update the search direction $\boldsymbol{p}_{k+1}$ for the next iteration and proceed until we reach a desired tolerance. We present the algorithm below in Algorithm 4.

Convergence

The convergence of Conjugate Gradient to the solution is typically linear and its speed is determined by the condition number $\kappa(A)$ of A, such that the larger the condition number is the slower the convergence will be. If the condition number is high, then we make use of preconditioning techniques in order to lower it.

Algorithm 4 Conjugate Gradient Method (CG)

function $\boldsymbol{x} \leftarrow$CG$(\boldsymbol{x}_0, A, \boldsymbol{b}, maxit, tol)$
  $\boldsymbol{r}_0 \leftarrow \boldsymbol{b} - A\boldsymbol{x}_0$
  $\boldsymbol{p}_0 \leftarrow \boldsymbol{r}_0$
  for $k \leftarrow 0 : maxit$ **do**
    Compute $\alpha_k \leftarrow \boldsymbol{r}_k^T \boldsymbol{r}_k / (\boldsymbol{p}_k^T A \boldsymbol{p}_k)$
    Compute $\boldsymbol{x}_{k+1} \leftarrow \boldsymbol{x}_k + \alpha_k \boldsymbol{p}_k$
    Compute $\boldsymbol{r}_{k+1} \leftarrow \boldsymbol{r}_k - \alpha_k \boldsymbol{p}_k$
    Check convergence:
    if $||\boldsymbol{r}_{k+1}|| < tol$ **then**
      break
    end if
    Compute $\beta_k \leftarrow \boldsymbol{r}_{k+1}^T \boldsymbol{r}_{k+1} / (\boldsymbol{r}_k^T \boldsymbol{r}_k)$
    Compute $\boldsymbol{p}_{k+1} \leftarrow \boldsymbol{r}_{k+1} + \beta_k \boldsymbol{p}_k$
  end for
  $\boldsymbol{x} \leftarrow \boldsymbol{x}_{k+1}$
end function

Preconditioned Conjugate Gradient (PCG)

In this section, we introduce some preconditioning techniques for the conjugate gradient method which aim to lower the condition number $\kappa(A)$ of the matrix A in the system $A\boldsymbol{x} - \boldsymbol{b} = 0$. We replace this system with the system

$$M^{-1}(A\boldsymbol{x} - \boldsymbol{b}) = 0 \tag{2.67}$$

where M is known as the preconditioner and chosen such that

$$\kappa(M^{-1}A) < \kappa(A).$$

Many preconditioning ideas have been proposed in the literature, see for example [42, 60, 13, 43, 88], and we will see some examples of appropriate preconditioners in later chapters. The modified method for preconditioned Conjugate Gradient is presented in Algorithm 5. we will see some examples of appropriate preconditioners in later chapters. The modified method for preconditioned Conjugate Gradient is presented in Algorithm 5.

We remark here that in more complex situations, a variable preconditioner may be used which changes at each iteration. This can cause a significant slow-down of the convergence when using the Fletcher-Reeves formula for β_k as above. Instead, the Polak-Ribière formula given by

$$\beta_k = \frac{\boldsymbol{z}_{k+1}^T (\boldsymbol{r}_{k+1} - \boldsymbol{r}_k)}{\boldsymbol{z}_k^T \boldsymbol{r}_k} \tag{2.68}$$

(where $\boldsymbol{z}_k = M^{-1}\boldsymbol{r}_k$) may provide a better convergence rate since it is locally optimal [88].

Algorithm 5 Preconditioned Conjugate Gradient Method (PCG)

function $\boldsymbol{x} \leftarrow$PCG$(\boldsymbol{x}_0, A, \boldsymbol{b}, M, maxit, tol)$
  $\boldsymbol{r}_0 \leftarrow \boldsymbol{b} - A\boldsymbol{x}_0$
  $\boldsymbol{z}_0 \leftarrow M^{-1}\boldsymbol{r}_0$
  $\boldsymbol{p}_0 \leftarrow \boldsymbol{z}_0$
  for $k \leftarrow 0 : maxit$ **do**
    Compute $\alpha_k \leftarrow \boldsymbol{r}_k^T \boldsymbol{z}_k / (\boldsymbol{p}_k^T A \boldsymbol{p}_k)$
    Compute $\boldsymbol{x}_{k+1} \leftarrow \boldsymbol{x}_k + \alpha_k \boldsymbol{p}_k$
    Compute $\boldsymbol{r}_{k+1} \leftarrow \boldsymbol{r}_k - \alpha_k \boldsymbol{p}_k$
    Check convergence:
    if $||\boldsymbol{r}_{k+1}|| < tol$ **then**
      break
    end if
    $\boldsymbol{z}_{k+1} \leftarrow M^{-1}\boldsymbol{r}_{k+1}$
    Compute $\beta_k \leftarrow \boldsymbol{z}_{k+1}^T \boldsymbol{r}_{k+1} / (\boldsymbol{z}_k^T \boldsymbol{r}_k)$
    Compute $\boldsymbol{p}_{k+1} \leftarrow \boldsymbol{z}_{k+1} + \beta_k \boldsymbol{p}_k$
  end for
  $\boldsymbol{x} \leftarrow \boldsymbol{x}_{k+1}$
end function

2.10.4 Additive Operator Splitting (AOS)

While explicit schemes such as time marching require very small time steps, leading to poor efficiency, the additive operator splitting scheme, introduced in [132, 203], is stable for larger time steps which can considerably improve the time required to find a solution. We consider a scheme of the form

$$u_t(t, \boldsymbol{x}) = f\left(u(t, \boldsymbol{x})\right) + \nabla \cdot \left(g \nabla u(t, \boldsymbol{x})\right) \tag{2.69}$$

with initial condition that at time $t = 0$ u is equal to the initial estimate, that is $u(0, \boldsymbol{x}) = u^{(0)}(\boldsymbol{x})$. Letting $\boldsymbol{x} = (x_1, \ldots, x_n)$, we can rewrite (2.69) as

$$u_t(t, \boldsymbol{x}) = f\left(u(t, \boldsymbol{x})\right) + \sum_{i=1}^{n} \left(g u_{x_i}(t, \boldsymbol{x})\right)_{x_i} = f\left(u(t, \boldsymbol{x})\right) + \left(g u_{x_1}(t, \boldsymbol{x})\right)_{x_1} + \ldots + \left(g u_{x_n}(t, \boldsymbol{x})\right)_{x_n}.$$

One-dimensional scheme

In one dimension, the equation we consider is given by

$$u_t = f(u) + \left(g u_x\right)_x.$$

Discretising with respect to time, we have

$$\frac{u^{k+1} - u^k}{\Delta t} = f\left(u^k\right) + \left(g u_x^{k+1}\right)_x$$

and further discretising with respect to space, we can write the discrete form of the above equation as

$$\begin{aligned}
&\frac{u_i^{k+1}-u_i^k}{\Delta t} \qquad (2.70)\\
&= f\left(u_i^k\right)+\nabla_x\left(g_i^k\nabla_x\left(u_i^{k+1}\right)\right)\\
&= f\left(u_i^k\right)+\nabla_x\left(g_i^k\frac{1}{h}\left(u_{i+1/2}^{k+1}-u_{i-1/2}^{k+1}\right)\right)\\
&\approx f\left(u_i^k\right)+\frac{1}{h}\nabla_x\left(\frac{g_{i+1/2}^k+g_{i-1/2}^k}{2}\left(u_{i+1/2}^{k+1}-u_{i-1/2}^{k+1}\right)\right) \qquad (2.71)\\
&= f\left(u_i^k\right)+\frac{1}{2h^2}\left(g_{i+1}^k+g_i^k\right)\left(u_{i+1}^{k+1}-u_i^{k+1}\right)-\frac{1}{2h^2}\left(g_i^k+g_{i-1}^k\right)\left(u_i^{k+1}-u_{i-1}^{k+1}\right)\\
&= f\left(u_i^k\right)+\frac{1}{2h^2}\left(g_{i+1}^k+g_i^k\right)\left(u_{i+1}^{k+1}-u_i^{k+1}\right)+\frac{1}{2h^2}\left(g_{i-1}^k+g_i^k\right)\left(u_{i-1}^{k+1}-u_i^{k+1}\right)\\
&= f\left(u_i^k\right)+\frac{1}{2h^2}\sum_{j\in\mathcal{N}(i)}\left(g_j^k+g_i^k\right)\left(u_j^{k+1}-u_i^{k+1}\right) \qquad (2.72)
\end{aligned}$$

where $\mathcal{N}(i)=\{i-1,i+1\}$ is the set of neighbours of i and (2.71) is due to setting g_i to be the average of its midpoint neighbours on either side. We can rearrange (2.72) to give

$$u_i^{k+1}-\frac{\Delta t}{2h^2}\sum_{j\in\mathcal{N}(i)}\left(g_j^k+g_i^k\right)\left(u_j^{k+1}-u_i^{k+1}\right)=u_i^k+\Delta t f\left(u_i^k\right). \qquad (2.73)$$

Now, since we have the following relationship

$$\sum_{j\in\mathcal{N}(i)}\left(g_j^k+g_i^k\right)\left(u_j^{k+1}-u_i^{k+1}\right)=\left(\sum_{j\in\mathcal{N}(i)}\left(g_j^k+g_i^k\right)u_j^{k+1}\right)-u_i^{k+1}\sum_{j\in\mathcal{N}(i)}\left(g_j^k+g_i^k\right)$$

we can rewrite (2.73) in matrix form as

$$Q^k\boldsymbol{u}^{k+1}=\boldsymbol{u}^k+\Delta t\boldsymbol{f}^k,\qquad Q^k=I-\frac{\Delta t}{2h^2}A^k \qquad (2.74)$$

where the matrix $A^k=\left[a_{i,j}^k\right]$ is given as

$$a_{i,j}^k=\begin{cases}-\sum_{j\in\mathcal{N}(i)}\left(g_j^k+g_i^k\right) & \text{if } j=i\\ g_j^k+g_i^k & \text{if } j\in\mathcal{N}(i)\\ 0 & \text{otherwise.}\end{cases}$$

So that A is a symmetric tridiagonal matrix. Without loss of generality, we can remove the $\boldsymbol{f}$ term and rewrite our system (2.74) as

$$\boldsymbol{u}^{k+1}=P^k\boldsymbol{u}^k,\qquad P^k=\left(Q^k\right)^{-1}=\left(I-\frac{\Delta t}{2h^2}A^k\right)^{-1}. \qquad (2.75)$$

For a given scheme of the type

$$u^{k+1} = P\left(u^k\right) u^k, \ \forall k \in \mathbb{N}, \qquad u^0 = f,$$

as we have above, we have to check that the following axioms are satisfied:

1. Continuity of the argument $P \in C\left(\mathbb{R}^N, \mathbb{R}^N \times \mathbb{R}^N\right)$. We show that P is strictly diagonally dominant. It is therefore invertible.

$$|P_{i,i}| = \left|1 + \Delta t \frac{g_{i-1} + g_i + g_{i+1}}{2h^2}\right| > \frac{\Delta t}{2h^2}\left(|g_i + g_{i+1}| + |g_i + g_{i-1}|\right) = \sum_{j \neq i} |P_{i,j}|.$$

2. Symmetry $p_{i,j} = p_{j,i}, \qquad \forall i, j \in J.$

3. Unit row sum $\sum_{j \in J} q_{i,j} = 1, \qquad \forall i \in J.$

4. Non-negativity $q_{i,j} \geq 0, \qquad \forall i, j \in J.$

5. Positive diagonal $q_{i,i} > 0, \qquad \forall i \in J.$

6. Irreducibility $\forall i, j \in J \ \exists k_0, \ldots, k_r \in J$ s.t. $k_0 = i, k_r = j, q_{k_\ell, k_{\ell+1}} \neq 0, \ \ell = 0, \ldots r-1.$

We solve this tri-diagonal system using the Thomas Algorithm described below.

The Thomas Algorithm

The aim is to solve a tridiagonal linear system given as

$$a_j u_j + b_j u_{j+1} + c_j u_{j+2} = l_j, \ j = 1, \ldots, N, \qquad u_0 = \alpha, \ u_{N+1} = \beta$$

which can be written in matrix form as

$$\begin{pmatrix} 1 & 0 & \cdots & \cdots & 0 \\ a_1 & b_1 & c_1 & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & a_N & b_N & c_N \\ 0 & \cdots & \cdots & 0 & 1 \end{pmatrix} \begin{pmatrix} u_0 \\ u_1 \\ \vdots \\ u_N \\ u_{N+1} \end{pmatrix} = \begin{pmatrix} \alpha \\ l_1 \\ \vdots \\ l_N \\ \beta \end{pmatrix}.$$

Gathering the boundary conditions, we can rewrite

$$a_1 u_0 + b_1 u_1 + c_1 u_2 = l_1 \ \Leftrightarrow \ b_1 u_1 + c_1 u_2 = l_1 - a_1 \alpha,$$

$$a_N u_{N-1} + b_N u_N + c_N u_{N+1} = l_N \ \Leftrightarrow \ a_N u_{N-1} + b_N u_N = l_N - c_N \beta,$$

and we may rewrite the system as

$$\begin{pmatrix} b_1 & c_1 & 0 & \cdots & 0 \\ a_2 & b_2 & c_2 & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & a_{N-1} & b_{N-1} & c_{N-1} \\ 0 & \cdots & 0 & a_N & b_N \end{pmatrix} \begin{pmatrix} u_1 \\ \vdots \\ \vdots \\ \vdots \\ u_N \end{pmatrix} = \begin{pmatrix} d_1 \\ \vdots \\ \vdots \\ \vdots \\ d_N \end{pmatrix}, \; d_j = \begin{cases} l_j - a_j\alpha & \text{if } j = 1 \\ l_j & \text{if } 1 < j < N \\ l_j - a_j\beta & \text{if } j = N. \end{cases}$$

Supposing that the first k equations have been written in the form

$$u_j + e_j u_{j+1} = f_j, \; j = 1, \ldots, k,$$

then we may write the kth and $k+1$th equations as

$$u_k + e_k u_{k+1} = f_k,$$

$$a_{k+1}u_k + b_{k+1}u_{k+1} + c_{k+1}u_{k+2} = d_{k+1}.$$

Writing the first as $u_k = f_k - e_k u_{k+1}$ and substituting into the second and rearranging, we have

$$u_{k+1} + e_{k+1}u_{k+2} = f_{k+1}, \qquad e_{k+1} = \frac{c_{k+1}}{b_{k+1} - a_{k+1}e_k}, \; f_{k+1} = \frac{d_{k+1} - a_{k+1}f_k}{b_{k+1} - a_{k+1}e_k}.$$

Therefore, we have

$$u_j + e_j u_{j+1} = f_j,$$

$$e_j = \begin{cases} \frac{c_j}{b_j} & \text{if } j = 1 \\ \frac{c_j}{b_j - a_j e_{j-1}} & \text{if } 1 < j \leq N \end{cases}, \quad f_j = \begin{cases} \frac{d_{j-1}}{b_j} & \text{if } j = 1 \\ \frac{d_j - a_j f_{j-1}}{b_j - a_j e_{j-1}} & \text{if } 1 < j \leq N. \end{cases}$$

Then, starting with $j = N$, we have

$$u_N + e_N u_{N+1} = f_N, , \quad e_N = \frac{c_N}{b_N - a_N e_{N-1}}, \quad f_N = \frac{d_N - a_N f_{N-1}}{b_N - a_N e_{N-1}}.$$

From this known u_N, we work backwards to successively obtain the remaining u_j.

Two dimensional scheme

Recall that the two-dimensional diffusion equation can be written in the form

$$u_t(t, \boldsymbol{x}) = f(u(t, \boldsymbol{x})) + \sum_{j=1}^{2} \left(g_j(u(t, \boldsymbol{x})) u_{x_j}(t, \boldsymbol{x}) \right)_{x_j}$$

with initial condition $u(0, \boldsymbol{x}) = u_0(\boldsymbol{x})$ and boundary condition $\partial u / \partial n = 0$ for $\boldsymbol{x} \in \partial\Omega$. We would like to discretise the equation. For the time discretisation, let Δt denote the step-size. Then we consider discrete time steps $t_k = k\Delta t$, $k \in \mathbb{N}$. Let h_l denote the grid step-size in the direction l. Regarding the two-dimension function u as a vector

with elements $\boldsymbol{u}_i = u(x_i)$, we let $u_i^k = u(x_i, t_k)$ and $g_i^k = g(u(x_i, t_k))$. We then give the discretisation of the diffusion equation as

$$\frac{u_i^{k+1} - u_i^k}{\Delta t} = \sum_{l=1}^{2} \sum_{j \in \mathcal{N}_l(i)} \frac{g_j^k + g_i^k}{2h_l^2} \left(u_j^{k+1} - u_i^{k+1} \right) + f(u_i^k).$$

As with the one-dimensional case, we may lose the term f without loss of generality in order to obtain the two-dimensional iterative scheme in matrix-vector notation as

$$\left(I - \Delta t \sum_{i=1}^{2} A_l(u) \right) u^{k+1} = u^k,$$

$$u^{k+1} = \left(I - \Delta t \sum_{i=1}^{2} A_l(u) \right)^{-1} u^k.$$

The following alternative method is proposed for the solution of the two dimensional problem

$$u^{k+1} = \frac{1}{2} \sum_{i=1}^{2} \left(I - 2\Delta t A_l \right)^{-1} u^k,$$

which may be solved efficiently using the Thomas Algorithm described above.

Chapter 3

Review of Variational Models for Image Reconstruction

There exist many approaches to tackling issues in image processing, such as filtering techniques for deblurring but our main concern is with the variational approach which has been shown to provide good results as well as interesting mathematical problems. It has also opened many ongoing research areas, much research has been carried out and is still ongoing.

3.1 Introduction

In this chapter, we introduce the main image restoration techniques which this thesis is concerned with, that is image deconvolution or deblurring. We also mention image denoising which is relevant in real-life applications. These problems are both important image pre-processing tasks since captured images are very rarely achieved with no noise and no blur. Even with advances in technology, blur remains an issue with many images in many areas such as consumer photography, astronomical imaging and medical imaging. The latter is our main concern. Blur is found in medical images from many different tools, including X-Ray and CT as well as in retinal imaging with Colour Fundus Photography and Fundus Angiography. The presence of blur can prevent medical grading, diagnosis and further processing techniques such as segmentation and inpainting, both of which can cope to a degree with noise but struggle to achieve good results with blurred images. Strong blurring of an image can render it useless, possibly meaning that a scan has to be repeated, causing further expense and delays in treatment.

Noise in an image is a very frequent problem with real data which gives rise to the need for denoising techniques which attempt to remove the noise from an image in order to leave a smoother version. Much research has been carried out which aims to do this efficiently while preserving the edges and details of the image. Once noise has been successfully removed, the image details may be seen clearly and the image may be ready for further processing. While it is possible to denoise an image before applying

a further technique such as deblurring this may be time consuming. It is therefore preferable to attempt to deblur an image while removing noise as a joint technique.

We now introduce the way in which we formulate such a variational technique, using denoising as an example since it can be easily explained. We begin by forming an energy functional using the available data and knowledge of the problem. In the case of denoising, the only available data is the received image given by the function $z(x, y)$. Making the assumption that the noise is additive, we aim to minimise the L^2-norm of the noise function $\eta(x, y) = u(x, y) - z(x, y)$ where u is the noiseless image. We therefore present the problem as

$$\min_u \left\{ f(u) = \frac{1}{2} \int_\Omega |u(x, y) - z(x, y)|^2 \, \mathrm{d}x \, \mathrm{d}y \right\} \tag{3.1}$$

which, since many possible functions u may satisfy this equation, can be regarded as ill-posed. We therefore introduce regularisation into (3.1) in an attempt to introduce stability and reduce the class of functions u which may minimise the functional. We present this modified problem as

$$\min_u \left\{ f(u) = \underbrace{\frac{1}{2} \int_\Omega |u(x, y) - z(x, y)|^2 \, \mathrm{d}x \, \mathrm{d}y}_{\text{fidelity term}} + \alpha \underbrace{J(u(x, y))}_{\text{regularisation term}} \right\} \tag{3.2}$$

where $J(u)$ provides the regularisation for the problem and $\alpha \in \mathbb{R}_{>0}$, known as the *regularisation parameter*, is a positive parameter which measures the trade-off between data fitting and regularisation. Two ongoing research areas concern the choice of parameter α and the choice of regularisation which may provide a better quality result.

The problem of deblurring, where the blur is assumed to be spatially invariant, may modelling in a similar manner with the blur taken into account. This is considered in more detail later sections.

3.2 Image Denoising

Noise is present in almost any acquired image and can be introduced at the acquisition stage and when storing the data. For example, noise may be introduced by poor illumination or inaccurate collection of the data. Noise varies considerably in magnitude, ranging from barely visible specks to heavy noise which is typical in astronomical imaging. Even small amounts of noise affect the perceived quality of an image and make fine details difficult to identify. Noise may also be dependent on signal intensity or independent of it. Below, we briefly classify a few of the many types of noise considered in image processing.

3.2.1 Gaussian Noise

Gaussian noise typically occurs during the acquisition of an image, for example by poor illumination. It is typically modelled as additive and independent of the signal intensity. Gaussian noise is also independent at each pixel and can vary between channels in colour images. The probability density function p is assumed to be that of the Gaussian distribution and given by

$$p(z) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(z-\mu)^2}{2\sigma^2}}$$

for a given grey level image (or colour image channel) z where μ is the mean value and σ is the standard deviation. Gaussian noise is said to be *white* if the mean value μ is equal to zero. Some examples of the effects of Gaussian noise introduced into a clean image can be seen in Figures 3.1–3.2.

3.2.2 Other Noise Types

We now briefly introduce some other noise types which are commonly met in image processing.

Poisson (Photon) Noise

In real imaging systems, photon noise and other sensor-based sources of noise contribute in varying proportions at different signal levels, leading to noise which is dependent on scene brightness. Understanding photon noise and modelling it explicitly is especially important for low-level computer vision tasks treating noisy images [[79, 129]] and for the analysis of imaging systems that consider different exposure levels [1, 97, 192] or sensor gains [96].

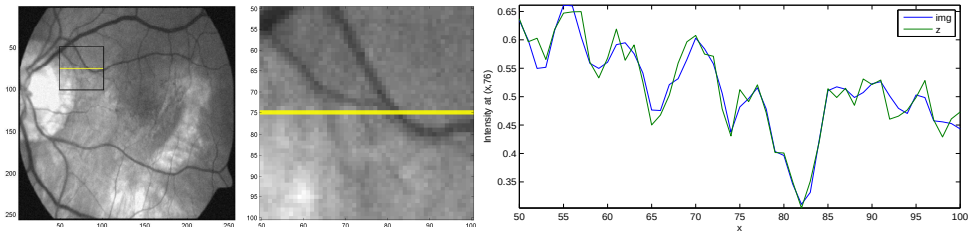
Photon counting is a classic Poisson process since individual photon detections can be treated as independent events that follow a random temporal distribution. The number N of photons measured by a given sensor element over a time interval t is described by the discrete probability distribution

$$Pr(N = k) = \frac{e^{-\lambda t}(\lambda t)^k}{k!}$$

where λ is the expected number of photons per unit time interval, which is proportional to the incident scene irradiance. This is a standard Poisson distribution with a rate parameter λt that corresponds to the expected incident photon count. The uncertainty described by this distribution is known as Poisson noise. Note that since the incident photon count follows a Poisson distribution, its variance is equal to its expectation $E[N] = \mathrm{Var}[N] = \lambda t$. From this, we see that Poisson noise is signal dependent and that its standard deviation increases with the square root of the signal.

Salt and Pepper Noise

Salt and pepper noise or *impulse* noise typically occurs as a result of transmission errors and can also be caused by errors in conversion between analogue and digital formats. It is characterised by dark pixels occurring in bright regions and bright pixels occurring in dark regions and can often be treated by interpolation involving nearby pixels.



(a) Noisy Image (b) Section of (a) (c) Intensity Values of Noisy Image compared with the Clean Image along Yellow Line

Figure 3.1: Example of the effect of adding Gaussian noise to give signal-to-noise ratio of 40. From left to right, we have (a) the noisy image, (b) the section of the noisy image outlined by the black square, (c) the intensity values of the noisy image along the yellow line (shown in green) compared with the intensity values of the clean, noiseless image along the same line (shown in blue).

3.2.3 Total Variation Denoising

The authors of [169] proposed a variational approach to the denoising problem of the form (3.2) which employed total variation (tv) regularisation due to it's improved performance over alternatives, such as the L^2-norm of the gradient. Their aim was to minimise the tv semi-norm of the image subject to the assumption that the noise is of zero-mean and standard deviation σ which is unknown. The problem is presented as

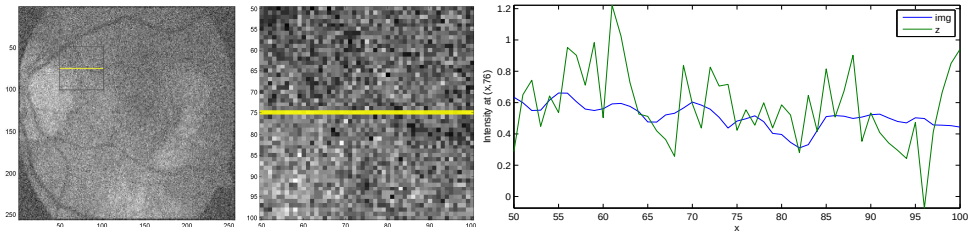
$$\min_u \left\{ f^{ROF}(u) = \frac{\alpha}{2} \int_\Omega |u(x,y) - z(x,y)|^2 \, \mathrm{d}x \, \mathrm{d}y + \int_\Omega |\nabla u(x,y)| \right\} \tag{3.3}$$

where $\alpha \in \mathbb{R}_{>0}$ is a small positive parameter which controls the amount of noise to be removed and the smoothness of the result. An important remark is that (3.3) is well-posed and we can guarantee the existence and uniqueness of the minimiser [40].

In order to find the minimiser u of the functional we derive the Euler-Lagrange equation given by

$$\begin{aligned} \alpha(u - z) - \nabla \cdot \left(\frac{\nabla u}{|\nabla u|} \right) &= 0 \text{ in } \Omega \\ \nabla u \cdot \nu &= 0 \text{ on } \delta\Omega \end{aligned} \tag{3.4}$$

where ν is the unit outward normal vector. There are many methods for solving this equation such as gradient descent as used in [169] or those presented in [82, 50, 173, 196].



(a) Noisy Image (b) Section of (a) (c) Intensity Values of Noisy Image compared with the Clean Image along Yellow Line

Figure 3.2: Example of the effect of adding Gaussian noise to give signal-to-noise ratio of 12. From left to right, we have (a) the noisy image, (b) the section of the noisy image outlined by the black square, (c) the intensity values of the noisy image along the yellow line (shown in green) compared with the intensity values of the clean, noiseless image along the same line (shown in blue).

It is also possible to minimise (3.3) directly as in [48].

Total variation denoising is popular since it can offer very impressive results. See Figure 3.3. However, it is also known for a phenomenon called the *staircasing effect* caused by the total variation transforming piecewise smooth functions into piecewise constant functions. This causes denoised images to look blocky. Much research has been carried out into finding alternative regularisation techniques which offer similar high-quality results regarding the noise reduction while avoiding the staircasing effect.

3.2.4 Alternative Regularisation

In an effort to improve on the good results of total variation, high-order models have been considered and are known for their ability to well-approximate smoother surfaces [39, 46, 133, 134, 202, 215]. Lysaker et. al [133] proposed the use of the regularisation term

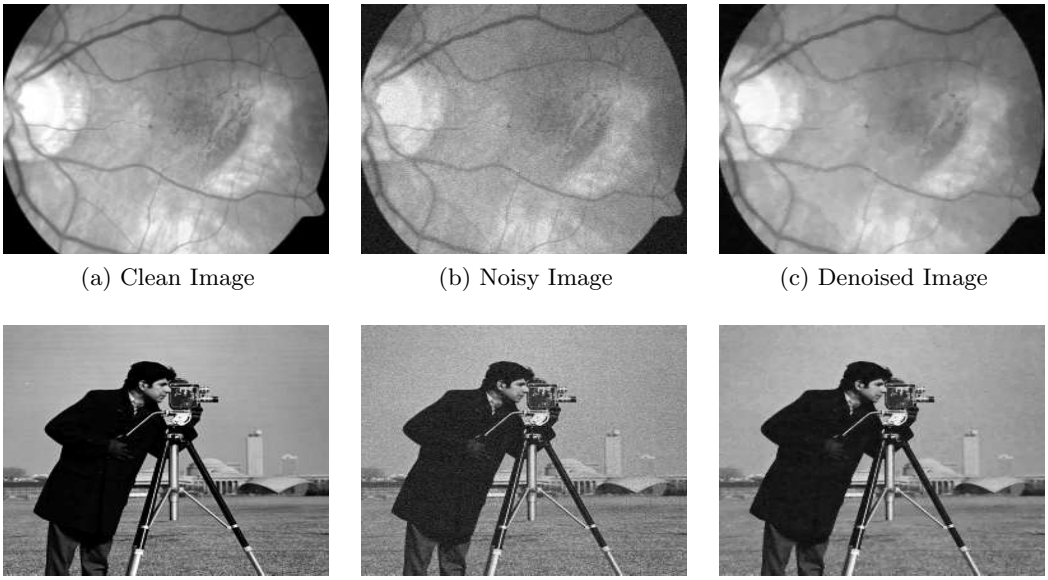
$$J(u) = \int_\Omega \sqrt{u_{xx}^2 + u_{xy}^2 + u_{yx}^2 + u_{yy}^2}\,\mathrm{d}x\,\mathrm{d}y.$$

When employed in a problem of the form (3.2), we aim to minimise the resulting functional by solving the associated Euler-Lagrange equation which is given by

$$\left(\frac{u_{xx}}{|D^2u|}\right)_{xx} + \left(\frac{u_{xy}}{|D^2u|}\right)_{yx} + \left(\frac{u_{yx}}{|D^2u|}\right)_{xy} + \left(\frac{u_{yy}}{|D^2u|}\right)_{yy} + \alpha(u-z) = 0$$

where $|D^2u| = \sqrt{u_{xx}^2 + u_{xy}^2 + u_{yx}^2 + u_{yy}^2}$. This formulation can yield excellent results on smooth regions where total variation regularisation can result in an approximation by piecewise constant functions.

Additionally, mixed models for regularisation have been proposed which aim to obtain good denoising results while preserving edges which high order models may struggle to do. Such models as those presented in [46, 39, 134, 56] attempt to combine different regularisation techniques in order to benefit from the edge-preserving properties of one



(a) Clean Image (b) Noisy Image (c) Denoised Image

(d) Clean Image (e) Noisy Image (f) Denoised Image

Figure 3.3: Illustration of the performance of total variation denoising to restore images from noise corruption. On the top row, we show an example of a Colour Fundus retina image (a) which has been corrupted by noise (b) with signal-to-noise-ratio (snr) of 28.1024 and restored using total variation (c) to snr 35.5171. On the bottom row, we show the camera-man example (a) which has been corrupted by noise (b) with snr of 28.1059 and restored using total variation (c) to snr 32.6741.

regularisation function and the smoothness properties of another. We discuss one such model in more detail in the context of deblurring in §3.3.6.

3.3 Image Deblurring

Blur in images is a very common problem, typically acquired at the acquisition stage and accompanied by noise. As well as being frequently present in consumer photography, blur is a major cause for image quality degradation with medical imaging. There are many contributing factors, including inaccurate focus of the camera, limited focal length and movement of the scene which is often unavoidable in medical imaging.

Blur affects images to varying degrees and clearly affects the quality of an image significantly, making edges less clear. Fine details are more difficult to see in blurred images, often to the extent that they are indistinguishable or hidden from view. This is particularly problematic in medical imaging where it is important to identify features accurately, including blood vessels which may be easily hidden by blur. Considering blur as spatial invariant, we may model the forward problem of obtaining the blurred image z from the ideal clean and sharp image u as

$$z(x, y) = [h * u](x, y) + \eta(x, y) \tag{3.5}$$

where h represents the blur function, η is the function describing the noise acquired

and $*$ represents the operation of convolution given by

$$[h * u](x, y) = \int_{-\infty}^{-\infty} \int_{-\infty}^{-\infty} h(x - x', y - y')u(x', y') \, \mathrm{d}x' \, \mathrm{d}y'.$$

The blur function h need not adhere to a particular structure, although it is usually considered to be positive and to have a unit integral. There are, however, some common patterns associated with certain blur types. Below, we classify some types of blur which are frequently met in image processing.

Out of Focus Blur

An image, or region of an image, is regarded as *in-focus* if the light from an object is well-converged. Objects which are in focus may appear sharp and clear. In contrast, if the light from an object is not well-converged, it is out-of-focus and can appear blurred. This can be caused by an incorrectly focussed lens or short depth of field, determined by the circle of confusion which gives the border between sharp and blurred points. This has the effect of blurring each pixel p with nearby pixels contained in a circle of a given range, centred at p. See Figures 3.4a and 3.4b for an example.

Gaussian Blur

Gaussian can appear optically and be used to approximate the blur caused by a lens. It is considered to be the result of blurring an image by a Gaussian function h which may written as

$$h(x, y) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right)$$

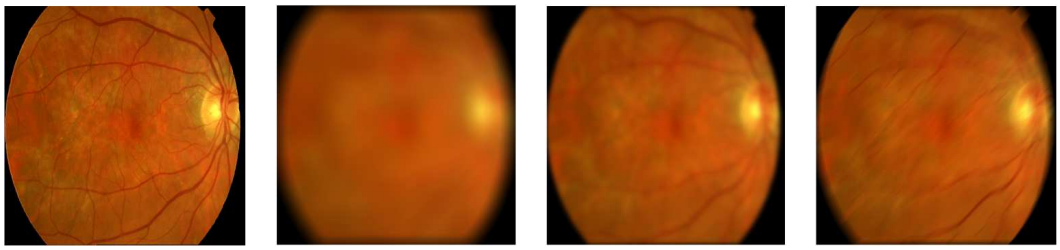
where σ is the standard deviation of the Gaussian distribution and the coefficient of the exponential term is chosen such that the integral of the function will be equal to one. In denoising, Gaussian blur can be introduced into an image by calculating the convolution of the noisy image with a Gaussian blur function. This provides an image with reduced noise but of course is not able to preserve sharp edges or fine details. See Figures 3.4a and 3.4c for an example.

Linear Motion Blur

Motion blur may be caused by movement of the camera or movement within the scene, for example by camera shake or a quick-travelling object. In particular, linear motion blur is that which travels in a single direction, the process of which can be visualised as the blurring of pixels along a straight line. See Figures 3.4a and 3.4d for an example.

3.3.1 Filtering Model

In this section, we introduce the Truncated Singular Value Decomposition (TSVD) model [193], which we derive from attempting to solve the equation (3.5) directly. We



(a) Retina Image u (b) Out of Focus (c) Gaussian (d) Linear Motion

Figure 3.4: Illustration of the effect of blur on an image. From left to right, we have (a) a Colour Fundus retina image u, (b) the same image u corrupted by out of focus blur, (c) the image u corrupted by Gaussian blur and (d) the image u corrupted by linear motion blur.

illustrate this problem in 1 dimension. We consider the Fredholm first kind integral equation of convolution type given by

$$z(x) = \int_0^1 k(x - x')u(x')\,\mathrm{d}x', \qquad 0 < x < 1, \tag{3.6}$$

where u represents the light source intensity, z represents the intensity of the image and k is the function describing the blur effects which occur during the capture of the image. The typical forward problem associated with (3.6) is to determine the blurred image z given the blur function k and the source u.

An associated inverse problem to this which is of interest to us is to determine the light source u given the blurred image z and the blur function k. The solution to this problem may appear to be simple. Indeed, we may discretise (3.6), obtaining the discrete system

$$\boldsymbol{z} = K\boldsymbol{u}, \tag{3.7}$$

and then calculate the discrete approximation $\boldsymbol{u} = K^{-1}\boldsymbol{z}$. However, if K is ill-conditioned then any errors in $\boldsymbol{z}$ will be amplified. Since these errors, such as those cause by noise in the received data, may not be controllable this method is not reliable.

We can however attempt to extract some useful information from system (3.7). To do this, we modify it to take into account noise

$$\boldsymbol{z} = K\boldsymbol{u}_{\text{true}} + \eta, \tag{3.8}$$

where η represents noise in the data and we write $\boldsymbol{u}_{\text{true}}$ as the true image simply to distinguish $\boldsymbol{u}$ as the approximated true image. Let $\delta = ||\eta||$ denote the error level which is typically unknown where $||\cdot||$ denotes the Euclidean norm. We will also assume that K is an invertible, real valued matrix. Then it has a singular value decomposition [87] given by

$$K = U\mathrm{diag}(s_i)V^T$$

where s_i are strictly positive decreasing values and we have the property that the right

singular vectors $\boldsymbol{v}_i$ of V and the left singular vectors $\boldsymbol{u}_i$ of U satisfy

$$\boldsymbol{u}_i^T \boldsymbol{u}_j = \delta_{i,j}, \qquad \boldsymbol{v}_i^T \boldsymbol{v}_j = \delta_{i,j}, \tag{3.9}$$

$$K\boldsymbol{v}_i = s_i \boldsymbol{u}_i, \qquad K^T \boldsymbol{u}_i = s_i \boldsymbol{v}_i, \tag{3.10}$$

where $\delta_{i,j}$ is the Kronecker delta, and the transposes of U and V are equal to their inverses. Using these properties, we can obtain an approximation to the true image given by

$$\boldsymbol{u} = V \mathrm{diag}(s_i^{-1}) U^T \boldsymbol{z} = \sum_{i=1}^{n} s_i^{-1} (\boldsymbol{u}_i^T \boldsymbol{z}) \boldsymbol{v}_i \tag{3.11}$$

where $\boldsymbol{u}_i$ and $\boldsymbol{v}_i$ are the column vectors of U and V respectively. An immediate observation is that (3.11) may involve division by many small values s_i, which has the effect of amplifying any noise or other error in the data. We would like to be able to retain some accuracy while reducing the impact of this issue.

To this end, we introduce a function $\omega_\alpha(s_i^2)$ which is multiplied my the values s_i to give

$$\boldsymbol{u} = V \mathrm{diag}(\omega_\alpha(s_i^2) s_i^{-1}) U^T \boldsymbol{z} = \sum_{i=1}^{n} \omega_\alpha(s_i^2) s_i^{-1} (\boldsymbol{u}_i^T \boldsymbol{z}) \boldsymbol{v}_i \tag{3.12}$$

where α is a parameter which may be chosen to suppress small singular values while retaining accuracy in the approximation. The function ω_α, known as the *filter function*, should be chosen such that large values of s_i are unchanged while the effect of small values is reduced. We present two commonly used filters below with the resulting models.

Truncated Singular Value Decomposition (TSVD)

We obtain this model by setting ω_α equal to zero for small values of s_i and equal to one for larger values, that is

$$\omega_\alpha(s_i) = \begin{cases} 1 & \text{if } s^2 > \alpha \\ 0 & \text{if } s^2 \leq \alpha \end{cases}$$

Equation (3.12) then reduces to

$$\boldsymbol{u} = \sum_{\substack{i=1 \\ s_i^2 > \alpha}}^{n} s_i^{-1} (\boldsymbol{u}_i^T \boldsymbol{z}) \boldsymbol{v}_i. \tag{3.13}$$

Tikhonov Filter Singular Value Decomposition (TFSVD)

Modifying our filter function ω_α to allow the model to retain some value of the smaller s_i and be close to one for larger values, we have the Tikhonov filter function

$$\omega_\alpha(s_i) = \frac{s_i^2}{s_i^2 + \alpha}$$

Equation (3.12) then reduces to

$$\boldsymbol{u} = \sum_{i=1}^{n} \frac{s_i(\boldsymbol{u}_i^T \boldsymbol{z})}{s_i^2 + \alpha} \boldsymbol{v}_i. \tag{3.14}$$

The parameter α is chosen by observation. If the parameter is too low, the noise filtering is not sufficient to obtain a good approximation. If it is chosen too high, then the noise can be filtered out but at a cost of losing details resulting in an overly smooth solution.

Such filter models are intuitive to understand and relatively straight-forward to encode.

3.3.2 Variational Approach

In this section, we introduce variational approaches to the problem of image deblurring. For very large ill-conditioned systems, it is not often practical to implement filtering models such as (3.12), (3.13) and (3.14) since they require the computation of the singular value decomposition of a large matrix.

We form the variational technique to tackle the deblurring problem by building an energy functional which makes use of the available data. Recall that we assume our received image to be the convolution of the hidden true data and a point spread function which describes the operation of the blur. We also acknowledge the possibility or indeed likelihood of the presence of noise $\eta = h * u - z$. We would like to minimise the L^2-norm of this, resulting in the problem

$$\min_u \left\{ f(u) = \frac{1}{2} \int_\Omega ([h * u](x,y) - z(x,y))^2 \, \mathrm{d}x \, \mathrm{d}y. \right\} \tag{3.15}$$

We may then attempt to minimise this functional by deriving the Euler Lagrange equation as follows. Letting, $\varepsilon \in \mathbb{R}$ be a real constant and $\phi(x,y)$ a function of x and y, we have

$$\begin{aligned}
\frac{\partial}{\partial \varepsilon} f(u + \varepsilon\phi)\bigg|_{\varepsilon=0} &= \frac{\partial}{\partial \varepsilon} \left(\frac{1}{2} \int_\Omega ([h * (u + \varepsilon\phi)](x,y) - z(x,y))^2 \, \mathrm{d}x \, \mathrm{d}y \right) \bigg|_{\varepsilon=0} \\
&= \frac{1}{2} \int_\Omega \frac{\partial}{\partial \varepsilon} ([h * u](x,y) + \varepsilon [h * \phi](x,y) - z(x,y))^2 \, \mathrm{d}x \, \mathrm{d}y \bigg|_{\varepsilon=0} \\
&= \int_\Omega ([h * u](x,y) + \varepsilon [h * \phi](x,y) - z(x,y))([h * \phi](x,y)) \, \mathrm{d}x \, \mathrm{d}y \bigg|_{\varepsilon=0} \\
&= \int_\Omega ([h * u](x,y) - z(x,y))([h * \phi](x,y)) \, \mathrm{d}x \, \mathrm{d}y \\
&= \int_\Omega h^\dagger(x,y) * ([h * u](x,y) - z(x,y))\phi(x,y) \, \mathrm{d}x \, \mathrm{d}y
\end{aligned} \tag{3.16}$$

where $h^\dagger(x,y) = h(-x,-y)$ is the adjoint of h and (3.16) is a consequence of the

following

$$\int [h * u - z](\boldsymbol{x})([h * \phi](\boldsymbol{x}))\,\mathrm{d}\boldsymbol{x} = \int [h * u - z](\boldsymbol{x}) \int h(\boldsymbol{x} - \boldsymbol{s})\phi(\boldsymbol{s})\,\mathrm{d}\boldsymbol{s}\,\mathrm{d}\boldsymbol{x}$$

$$= \int\int [h * u - z](\boldsymbol{x})h(\boldsymbol{x} - \boldsymbol{s})\,\mathrm{d}\boldsymbol{x}\phi(\boldsymbol{s})\,\mathrm{d}\boldsymbol{s} = \int\int [h * u - z](\boldsymbol{x})h^{\dagger}(\boldsymbol{s} - \boldsymbol{x})\,\mathrm{d}\boldsymbol{x}\phi(\boldsymbol{s})\,\mathrm{d}\boldsymbol{s}$$

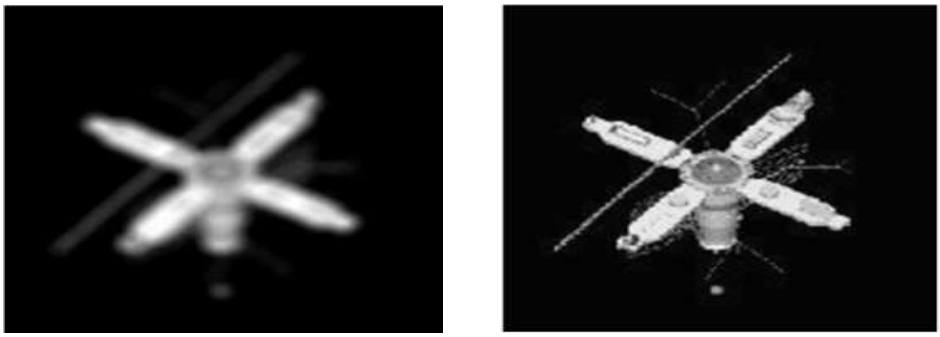
$$= \int \Big[h^{\dagger}(\boldsymbol{s} - \boldsymbol{x}) * (h * u - z)\Big](\boldsymbol{s})\phi(\boldsymbol{s})\,\mathrm{d}\boldsymbol{s}$$

where $\boldsymbol{x} = (x, y)$ and $\boldsymbol{s} = (s, t)$. It follows that the associated Euler-Lagrange equation for the functional f given by (3.15) may be presented as

$$h^{\dagger}(x, y) * ([h * u](x, y) - z(x, y)) = 0. \tag{3.17}$$

Given an image z which contains no noise, that is the received data is precisely equal to the convolution of a blur function with the true image, the solution to (3.17) which minimises (3.15) can give a good approximation of the true image. See Figure 3.5.

The inevitable presence of noise makes this problem ill-posed so that the solution to (3.17) is unlikely to be the true image, as can be seen in Figure 3.6. We therefore introduce regularisation in order to make the problem well-posed. We describe below some commonly used regularisation functions.



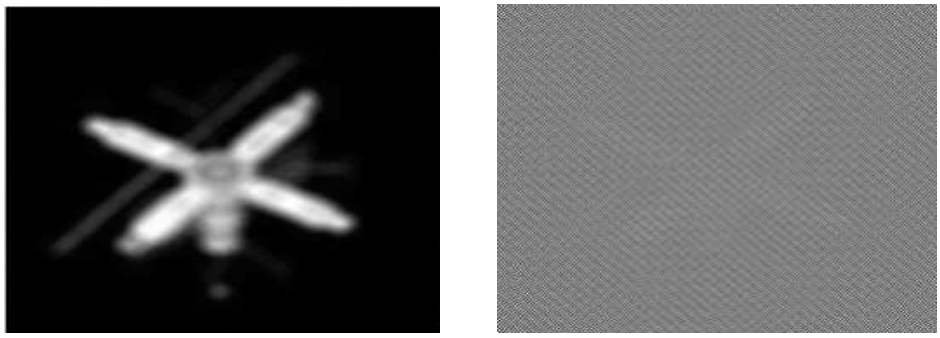
(a) Received Image (b) Restored Image

Figure 3.5: Illustration of the performance of (3.15) with blurred data which is free of noise. It can be noted that the restoration (b) of the image (a) with no additive noise by the minimisation (3.15) yields a very close approximation to the true solution.

3.3.3 Tikhonov Regularised Deblurring

We now introduce the Tikhonov regularisation function which aims to limit the sum of the squared intensity values. As such, this is presented as the problem

$$\min_{u} \left\{ f^{Tik}(u) = \frac{1}{2} \int_{\Omega} ([k * u](\boldsymbol{x}) - z(\boldsymbol{x}))^2\,\mathrm{d}\boldsymbol{x} + \frac{\alpha}{2} \int_{\Omega} |u(\boldsymbol{x})|^2\,\mathrm{d}\boldsymbol{x} \right\}. \tag{3.18}$$



(a) Received Image (b) Restored Image

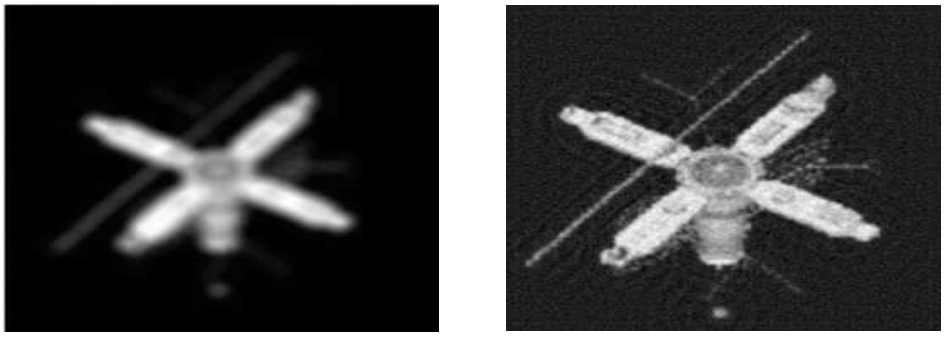
Figure 3.6: Illustration of the performance of (3.15) with noisy data. The blurred image (a) has been achieved by adding a small amount of noise to Figure 3.5a. It can be noted that, although the noise is visually imperceptible, the presence of noise means that the image cannot be restored (b) by the minimisation (3.15).

The function f^{Tik} is minimised by the solution to

$$h^\dagger(\boldsymbol{x}) * (h(\boldsymbol{x}) * u(\boldsymbol{x}) - z(\boldsymbol{x})) + \alpha u(\boldsymbol{x}) = 0. \tag{3.19}$$

By the associativity and distributivity properties of convolution, (3.19) can be rewritten as

$$\Big(h^\dagger(\boldsymbol{x}) * h(\boldsymbol{x}) + \alpha\Big) * u(\boldsymbol{x}) = h^\dagger(\boldsymbol{x}) * z(\boldsymbol{x}).$$



(a) Received Image (b) Restored Image

Figure 3.7: Illustration of the performance of (3.18) with noisy data. The noisy and blurred image (a), which is the same as the image used in Figure 3.6a, has been restored (b) by the minimisation (3.18) with $\alpha = 10^{-3}$. It can be noticed that the restored image (b) is a considerable improvement on the result of restoring (a) by the minimisation (3.15) shown in Figure 3.6b. (b) is much sharper and hidden details are revealed.

3.3.4 L^2 Regularised Deblurring

We can notice that while Figure 3.7b is a considerable improvement, many defects are visible in the restored image. We attempt to reduce such affects by modifying our constraint to minimise the sum of the squared gradient. The aim here is to minimise

defects, providing a smoother resulting image. Introducing this constraint into the function, we present the modified problem as

$$\min_u \left\{ f^{L^2}(u) = \frac{1}{2}\int_\Omega ([k * u](\boldsymbol{x}) - z(\boldsymbol{x}))^2 \,\mathrm{d}\boldsymbol{x} + \frac{\alpha}{2}\int_\Omega |\nabla u(\boldsymbol{x})|^2 \,\mathrm{d}\boldsymbol{x} \right\}. \tag{3.20}$$

In order to calculate the associated Euler-Lagrange equation, we find the derivative of the regularisation function given by

$$\begin{aligned}
\frac{\partial}{\partial\varepsilon} J(u+\varepsilon\phi)\bigg|_{\varepsilon=0} &= \frac{\partial}{\partial\varepsilon}\frac{1}{2}\int_\Omega |\nabla(u+\varepsilon\phi)|^2 \,\mathrm{d}\boldsymbol{x}\bigg|_{\varepsilon=0} = \frac{1}{2}\int_\Omega \frac{\partial}{\partial\varepsilon}|\nabla u + \varepsilon\nabla\phi|^2 \,\mathrm{d}\boldsymbol{x}\bigg|_{\varepsilon=0} \\
&= \int_\Omega (\nabla u + \varepsilon\nabla\phi)\frac{\partial}{\partial\varepsilon}(\nabla u + \varepsilon\nabla\phi)\,\mathrm{d}\boldsymbol{x}\bigg|_{\varepsilon=0} = \int_\Omega \nabla u\nabla\phi \,\mathrm{d}\boldsymbol{x} \\
&= \int_\Omega \nabla\cdot\nabla u\phi \,\mathrm{d}\boldsymbol{x} = \int_\Omega \Delta u\phi \,\mathrm{d}\boldsymbol{x}
\end{aligned} \tag{3.21}$$

where the fifth equality of (3.21) is given by the relation

$$\int_\Omega F\cdot\nabla v \,\mathrm{d}\boldsymbol{x} = \int_\Gamma Fv\cdot\tilde{n}\,\mathrm{d}\boldsymbol{x} - \int_\Omega \nabla\cdot Fv\,\mathrm{d}\boldsymbol{x}.$$

From this, we can write the following Euler-Lagrange equation which minimises (3.20):

$$k^\dagger(\boldsymbol{x}) * (k(\boldsymbol{x}) * u(\boldsymbol{x}) - z(\boldsymbol{x})) - \alpha\Delta u(\boldsymbol{x}) = 0. \tag{3.22}$$

Considering the discrete counterpart of the regularisation term Δu over a mesh of step-size h_x and h_y, we calculate the Laplacian of the discrete data u using finite differences. At each point $u_{i,j}$

$$\Delta(u_{i,j}) = \nabla^x_-(\nabla^x_+(u_{i,j})) + \nabla^y_-(\nabla^y_+(u_{i,j})) \tag{3.23}$$

where the derivatives are given by

$$\nabla^x_\pm(u_{i,j}) = \frac{\mp 1}{h_x}\left(u_{i,j} - u_{i,j\pm1}\right), \qquad \nabla^y_\pm(u_{i,j}) = \frac{\mp 1}{h_y}\left(u_{i,j} - u_{i\pm1,j}\right).$$

The stencil for the Laplacian of u (3.23) can be given as

$$\mathcal{S} = \frac{1}{h_x^2 h_y^2}\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}.$$

It can be observed then that the (3.23) may be written as the discrete convolution of u with the associated matrix of $\mathcal{S}$ given by

$$S = \frac{1}{h_x^2 h_y^2}\begin{pmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{pmatrix}.$$

The associated discrete problem of (3.22) can then be written as

$$(k^\dagger \odot k - \alpha S) \odot u = k^\dagger \odot z \tag{3.24}$$

where $\odot$ represents the operation of discrete convolution.

3.3.5 Total Variation Deblurring

The authors of [169] aim to minimise the total variation semi-norm of the image due to its ability to remove noise from the image without smoothing edges. Due to this property, it is a popular regularisation function for image deblurring. Incorporating this into the deblurring model, we present the total variation deblurring model as

$$\min_u \left\{ f^{TV}(u) = \frac{1}{2} \int_\Omega ([k * u](\boldsymbol{x}) - z(\boldsymbol{x}))^2 \, \mathrm{d}\boldsymbol{x} + \alpha \int_\Omega |\nabla u(\boldsymbol{x})| \right\}. \tag{3.25}$$

which has Euler-Lagrange equation

$$k^\dagger(\boldsymbol{x}) * (k(\boldsymbol{x}) * u(\boldsymbol{x}) - z(\boldsymbol{x})) - \alpha \nabla \cdot \left(\frac{\nabla u(\boldsymbol{x})}{|\nabla u(\boldsymbol{x})|} \right) = 0. \tag{3.26}$$

3.3.6 Alternative Regularisation Models

Similar to the case of denoising, many efforts have been made to improve the quality of deblurring results over total variation by using high order and mixed regularisation methods. See for example [56, 39, 46, 133, 134, 202, 215].

Chang et. al. [56] proposed using a convex combination of the total variation model of Rudin, Osher and Fatemi (ROF) [169] and the fourth order method by Lysaker, Lundervold and Tai (LLT) [133] described in §3.2.4. Since the ROF model is known to give improved results over LLT in regions with discontinuities, and LLT gives improved results for smooth signals, the authors of [56] aim to take advantage of both of these properties by introducing a control parameter θ which is dependent on the local absolute gradient of the image. The problem is presented as

$$\min_u \left\{ f^{CTX}(u) = \alpha \left(\int_\Omega \theta |\nabla u| \, \mathrm{d}\boldsymbol{x} + \int_\Omega (1-\theta) |D^2 u| \, \mathrm{d}\boldsymbol{x} \right) + \frac{1}{2} \int_\Omega (k*u - z)^2 \, \mathrm{d}\boldsymbol{x} \right\}, \tag{3.27}$$

where k is the blur function, u the image, z the received image, α is a tuning parameter which measures the trade-off between data fitting and regularisation, ∇u is the gradient of u and the term $|D^2 u|$ is given by

$$|D^2 u| = \sqrt{(u_{xx})^2 + (u_{xy})^2 + (u_{yx})^2 + (u_{yy})^2}.$$

The aim is to select the parameter θ such that the regularisation component of the model will be approximately equal to total variation in regions with low and high gradients and approximately equal to the LLT regularisation term in smooth regions with lower gradients. Additionally, ROF is still favoured for very low gradients due

to its ability to well approximate such regions. In order to obtain this automatically, the parameter θ is given as a piecewise function depending on the image intensity and some additional parameters c_0, c_1, c_d (which are set to $(c_0, c_1, c_d) = (0, 40, 0.1)$ in [56]):

$$\theta = \begin{cases} 1 & \text{if } |\nabla u| \leq c_0 \text{ and } |\nabla u| \geq c_1 \\ c_d & \text{if } c_0 + 5 \leq |\nabla u| \leq c_1 - 5 \\ 1 - \frac{(|\nabla u| - c_d)(1-c_d)}{5} & \text{if } c_0 \leq |\nabla u| \leq c_0 + 5 \\ 1 + \frac{(|\nabla u| - c_1)(1-c_d)}{5} & \text{if } c_1 - 5 \leq |\nabla u| \leq c_1. \end{cases} \tag{3.28}$$

Since the parameter θ is a function which is dependent on u, the Euler-Lagrange equation for the minimisation problem (3.27) can become quite complicated. The authors therefore propose an iterative solution, calculating θ at each step and regarding it as a fixed parameter when minimising f^{CTX} of (3.27). At a particular step i, this yields the Euler Lagrange equation

$$\alpha\Bigg(\left(\frac{\left(1-\theta^i\right) u_{xx}^{i+1}}{|D^2 u^{i+1}|} \right)_{xx} + \left(\frac{\left(1-\theta^i\right) u_{xy}^{i+1}}{|D^2 u^{i+1}|} \right)_{yx} + \left(\frac{\left(1-\theta^i\right) u_{yx}^{i+1}}{|D^2 u^{i+1}|} \right)_{xy} + \left(\frac{\left(1-\theta^i\right) u_{yy}^{i+1}}{|D^2 u^{i+1}|} \right)_{yy}$$

$$- \left(\left(\frac{\left(\theta^i\right) u_x^{i+1}}{|Du^{i+1}|} \right)_x + \left(\frac{\left(\theta^i\right) u_y^{i+1}}{|Du^{i+1}|} \right)_y \right) \Bigg) + k^\dagger * \left(k * u^{i+1} - z \right) = 0, \tag{3.29}$$

where $|Du| = \sqrt{(u_x)^2 + (u_y)^2}$.

Iterating on i, at a general step, we solve (3.29) for u^{i+1} given θ^i and then calculate θ^{i+1} given u^{i+1} using (3.28). In this way, θ should be successively better chosen at each iteration i yielding a result which has sharp edges and smooth regions elsewhere.

3.3.7 Deblurring in the Presence of Poisson Noise

Many areas of imaging, including medical imaging, suffer not only from additive white Gaussian noise but from Poisson (or photon) noise which often contributes in varying proportions to the noise in the image. If we assume that the noise follows a Poisson distribution, then we can estimate the image as the maximiser of the probability of acquiring the image given the received image [66, 182]. The probability of $k \in \mathbb{N}$ photon impacts complies with the Poisson distribution given by

$$P_\lambda(X = k) = \frac{\lambda^k}{k!} e^{-\lambda}$$

where $\lambda \in \mathbb{R}_{>0}$ is both the mean and the variance of the distribution. The probability of of acquiring an image z given that the true object is u is [66]

$$p(z|u) = \prod_{\boldsymbol{x} \in \Omega} \left(\frac{([h * u](\boldsymbol{x}))^{z(\boldsymbol{x})}}{z(x)!} \exp -[h * u](\boldsymbol{x}) \right). \tag{3.30}$$

The original image can then be estimated by a maximum-likelihood approach as the maximiser of the probability with respect to the image z.

Richardson Lucy

Taking the negative logarithm of (3.30) we achieve the functional whose minimiser u is an approximation to the true image [182]:

$$f^{RL}(u) = \int_\Omega \left(h * u - z - z \ln \left(\frac{h * u}{z} \right) \right) \mathrm{d}\boldsymbol{x}. \tag{3.31}$$

In order to find a solution, we calculate the minimisation in order to derive the Euler-Lagrange equation as follows

$$\begin{aligned}
\frac{\partial}{\partial \varepsilon} f(u + \varepsilon\phi)\Big|_{\varepsilon=0} &= \frac{\partial}{\partial \varepsilon} \int_\Omega \left(h * (u + \varepsilon\phi) - z - z \ln \left(\frac{h * (u + \varepsilon\phi)}{z} \right) \right) \mathrm{d}\boldsymbol{x} \Big|_{\varepsilon=0} \\
&= \int_\Omega \frac{\partial}{\partial \varepsilon} \left(h * u + \varepsilon h * \phi - z - z \ln \left(\frac{h * u + \varepsilon h * \phi}{z} \right) \right) \mathrm{d}\boldsymbol{x} \Big|_{\varepsilon=0} \\
&= \int_\Omega \left(h * \phi - z \left(\frac{z}{h * u + \varepsilon h * \phi} \right) \frac{\partial}{\partial \varepsilon} \left(\frac{h * u + \varepsilon h * \phi}{z} \right) \right) \mathrm{d}\boldsymbol{x} \Big|_{\varepsilon=0} \\
&= \int_\Omega \left(h * \phi - z \left(\frac{z}{h * u + \varepsilon h * \phi} \right) \left(\frac{h * \phi}{z} \right) \right) \mathrm{d}\boldsymbol{x} \Big|_{\varepsilon=0} \\
&= \int_\Omega \left(h * \phi - z \left(\frac{z}{h * u} \right) \left(\frac{h * \phi}{z} \right) \right) \mathrm{d}\boldsymbol{x} \\
&= \int_\Omega h * \phi \left(1 - \left(\frac{z}{h * u} \right) \right) \mathrm{d}\boldsymbol{x} \\
&= \int_\Omega \left(h^\dagger * \left(1 - \left(\frac{z}{h * u} \right) \right) \right) \phi \, \mathrm{d}\boldsymbol{x}
\end{aligned} \tag{3.32}$$

since

$$\int [h * \phi](\boldsymbol{x}) g(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x} = \int \int h(\boldsymbol{x} - \boldsymbol{s}) \phi(\boldsymbol{s}) \, \mathrm{d}\boldsymbol{s} \, g(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x} = \int \int h(\boldsymbol{x} - \boldsymbol{s}) \phi(\boldsymbol{s}) g(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{s} \, \mathrm{d}\boldsymbol{x}$$
$$= \int \int h(\boldsymbol{x} - \boldsymbol{s}) g(\boldsymbol{x}) \phi(\boldsymbol{s}) \, \mathrm{d}\boldsymbol{x} \, \mathrm{d}\boldsymbol{s} = \int \int h^\dagger(\boldsymbol{s} - \boldsymbol{x}) g(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x} \, \phi(\boldsymbol{s}) \, \mathrm{d}\boldsymbol{s} = \int \left[h^\dagger * g \right] (\boldsymbol{s}) \, \phi(\boldsymbol{s}) \, \mathrm{d}\boldsymbol{s}.$$

where $h^\dagger(\boldsymbol{x}) = h(-\boldsymbol{x})$ is the adjoint of the point spread function h. This results in the Euler Lagrange equation

$$h^\dagger * \left(1 - \left(\frac{z}{h * u} \right) \right) = 0.$$

By instead calculating the Euler-Lagrange equation using a multiplicative step $u(1+\varepsilon\phi)$ rather than an additive one, and adopting a fixed point iteration scheme we achieve the well-known Richardson-Lucy method [167, 66] given by carry out iterations using

$$u^{k+1} = \left(h^\dagger * \frac{z}{h * u^k} \right) u^k. \tag{3.33}$$

Note that we have made use of the relation $h^\dagger * 1 = 1$ to obtain the left hand side of (3.33).

This method can achieve good results where the received data can be modelled as $z = h * u$, that is without noise. However, it is known that this scheme may diverge as the iteration count tends to infinity. We require therefore a method of stabilising the result in the presence of noise. This is considered below.

Regularised Richardson Lucy

A common approach in variational modelling to offset the effects that noise in received data can have on the solution is to introduce constraints on the smoothness of the solution in the form of regularisation. As with [66, 174], regularisation may be introduced to extend (3.31) in order to obtain the functional

$$f^{RRL}(u) = \int_{\Omega} \left(h * u - z - z \ln \left(\frac{h * u}{z} \right) + \alpha \Psi \left(|\nabla u|^2 \right) \right) \mathrm{d}\boldsymbol{x}. \tag{3.34}$$

where α is a regularisation parameter which may be adjusted to change the amount of smoothness to fit the problem, and Ψ determines the regularisation equation to be used. There are many common options for Ψ such as $\Psi(s^2) = \sqrt{s^2 + \beta}$ to give the smooth approximation to the total variation semi-norm and the Charbonnier [57] term $\Psi'(s^2) = (1 + s^2/\beta^2)^{-1/2}$ to name but two. Similar to the above case, we can minimise this model with respect to the image by deriving the Euler-Lagrange equation:

$$h^{\dagger} * \left(1 - \left(\frac{z}{h * u} \right) \right) - \alpha \nabla \cdot \left(\Psi' \left(|\nabla u|^2 \right) \nabla u \right) = 0.$$

This can also be solved in a fixed point manner by calculating the minimisation of the functional (3.34) using a multiplicative step and rearranging the equation. Care must be taken however with the choice of steps in the iterative scheme. This will be considered in the next section along with a generalisation of this model.

Robust Regularised Richardson-Lucy

The idea of using robust statistics has also been included in the Richardson Lucy model [204, 105] by introducing a non-negative increasing function $\Phi : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ into the fitting term in order to gain robustness against outliers and imprecise data. The revised robust model is presented as

$$f^{RRRL}(u) = \int_{\Omega} \left(\Phi \left(h * u - z - z \ln \left(\frac{h * u}{z} \right) \right) + \alpha \Psi \left(|\nabla u|^2 \right) \right) \mathrm{d}\boldsymbol{x}. \tag{3.35}$$

A particular choice for the additional function which is known to produce good results is $\Phi(s) = 2\sqrt{s + \beta}$ for small, non-negative $\beta \in \mathbb{R}_{>0}$. Note that choosing the function $\Phi(s) = s$ reduces the functional to the regularised model (3.34) above. The Euler Lagrange equation can be derived as

$$h^{\dagger} * \left(\Phi'(f_s(u)) \left(1 - \left(\frac{z}{h * u} \right) \right) \right) - \alpha \nabla \cdot \left(\Psi' \left(|\nabla u|^2 \right) \nabla u \right) = 0$$

where $f_s(u) = h * u - z - z \ln\left(\frac{h*u}{z}\right)$.

In order to derive a solution strategy, the authors of [66] propose a method using fixed point iterations for the case $\Phi(s) = s$ which allows for the fitting term to be split into one component which will be updated at the current iteration step and another which will be updated at a later step. Similarly, the calculation of the regularisation term is lagged, yielding the iteration scheme given by

$$u^{k+1} = \frac{u^k}{1 - \alpha\nabla \cdot \left(\Psi'\left(|\nabla u^k|^2\right)\nabla u^k\right)}\left(h^\dagger * \frac{z}{h * u^k}\right).$$

While this method can work quickly, it is clear that the value of α must be restricted in order to avoid singularities at an iteration step or introducing negative values into the restored image. The authors of [204] present a method to overcome this problem for the general case, by splitting the regularisation term into positive and negative components

$$R_\pm(u) = \frac{1}{2}\left(R(u) \pm |R(u)|\right), \qquad R(u) = \nabla \cdot \left(\Psi'\left(\left|\nabla u^k\right|^2\right)\nabla u^k\right)$$

in order to obtain the more robust scheme

$$u^{k+1} = \frac{\left(h^\dagger * \left(\Phi'\left(f_s\left(u^k\right)\right)\frac{z}{h*u^k}\right)\right) + \alpha R_+\left(u^k\right)}{h^\dagger * \Phi'\left(f_s\left(u^k\right)\right) - \alpha R_-\left(u^k\right)}u^k.$$

This allows for all of the components of the denominator of the expression to always be positive and imposes non-negativity but experiments have shown that the model struggles to obtain good results under heavy regularisation [157]. In order to better stabilise this, [157] introduces an alternative fixed point scheme which also allows for acceleration by avoiding the split of the convolution with the adjoint of the point spread function. Applying the steepest descent method, the iterative scheme is presented as

$$\frac{u^{k+1} - u^k}{\delta_t} = h^\dagger * \left(\Phi'\left(f_s\left(u^k\right)\right)\left(1 - \frac{z}{h * u^k}\right)\right)u^{k+1} - \alpha\nabla \cdot \left(\frac{\nabla u^{k+1}}{|\nabla u^k|}\right)u^k, \tag{3.36}$$

where δ_t is the step-size which may be chosen to be relatively large.

3.3.8 Semi-Blind Models

Semi-blind deblurring [2, 11] involves recovering a hidden true image with only partial knowledge or some assumptions about the blur function, such as the type of blur [123, 32]. Such models perform well and can obtain improved results over blind deblurring when the blur type may be known or estimated. Such techniques are useful in related areas such as the segmentation of blurred images [11] and super-resolution [130, 217] where the blur is often of Gaussian or out of focus type. Parametric deblurring is a particular type of semi-blind deblurring which assumes that we may be able to model the blur degradation term h as a parametric function dependent on only a few parameters. Our aim in this case is to recover the parameters and thus reconstruct

the blur function.

Intensity-constraints in deblurring models [13, 15, 44, 182, 181] have become popular due to their ability to avoid intensity values falling outside of their expected range, which can cause poor results, particularly in astronomical imaging, medical imaging and blind and semi-blind deconvolution. We present a parametric model which incorporates intensity constraints implicitly for the image. This is found to enhance the quality of results when compared to unconstrained models.

Much research involving parametric models either assumes that Gaussian blur is the cause for degradation or models alternative blur functions as piecewise constant functions. Assuming that the blur is of Gaussian type is a limitation which may prevent the accurate restoration of images corrupted by other blur types, so we would like to extend this to accommodate other blurs. We note that modelling a blur function as a piecewise constant prohibits the recovery of the parameters within a variational framework.

Many papers present techniques which may be classed as semi-blind in the sense that their aim is to restore images which have been corrupted by blur of a particular type [32, 123, 165, 179, 214]. [123] presents a method for blind deconvolution assuming that the degradation in the image is caused by motion blur. While not presented as a parametric model, the authors attempt to find the length or blurring distance of the blur function as shown in Equation (3.37) by using image statistics. [32] also presents a non-parametric model which assumes that the degradation is caused by motion blur, and makes use of multiple images to estimate the precise blur function.

Many papers attempt to improve the estimation of point spread functions by considering them as piecewise functions after discretisation. This reduces the estimation of many unknowns to that of only a few variables which determine the entire psf.

Semi-blind deblurring using piecewise representations of blur functions

In [18], the authors give several equations for piecewise constant representations of kernel functions. Motion blur, caused by relative motion between the subject and the imaging device, is presented in the case of constant horizontal velocity translation [18, 102] as

$$h(x,y;s,t) = h(x-s) = \begin{cases} \frac{\delta(y-t)}{VT} & \text{for } 0 \leq x-s \leq VT \\ 0 & \text{otherwise,} \end{cases}$$

where V is the measure of the velocity and $[0,T]$ is the exposure interval. The discrete equivalent uses the blurring distance L and is thus presented as

$$h(i,j;k,l) = h(i-k) = \begin{cases} \frac{1}{L+1} & \text{for } 0 \leq i-k \leq L,\ j=l \\ 0 & \text{otherwise.} \end{cases} \tag{3.37}$$

Out of Focus blur, which occurs when some parts of the scene are in focus and others are not, concerns the intensity distribution within the circle of confusion of radius r around a point. The corresponding point spread function is presented [18, 102] as

$$h(x,y) = \begin{cases} \frac{1}{\pi r^2} & \text{for } x^2 + y^2 \leq r^2 \\ 0 & \text{otherwise.} \end{cases}$$

The parameters are estimated from the power spectrum of the blur function by approximating the power spectrum of the true image using a wide variety of images [186].

[78] makes use of semi-blind deconvolution by reducing the number of unknown variables in the PSF to a small number of variables from which it can be given in full. [117] also presents discrete piecewise equations for linear motion blur and out of focus blur, making use of frequency domain zeros to influence the identification of the parameters.

[9] presents discrete equations for several blur types including box blur, also known as uniform 2D blur, as

$$h(i,j) = \begin{cases} \frac{1}{L^2} & \text{for } -\frac{L}{2} \leq i,j \leq \frac{L}{2} \\ 0 & \text{otherwise,} \end{cases}$$

where L is assumed to be an odd integer. This can be viewed as a composite motion blur in horizontal and vertical directions.

Recently, [130] and [217] have made use of parametric formulations of blur functions for super-resolution techniques. The parameters are estimated prior to deblurring in cases where it is known or assumed that the blur function may be of Gaussian or out-of-focus type.

Semi-blind deblurring using continuous representations of the blur function

Gaussian blur has a well-known formulation given by Equation (3.38). Recently, papers aimed at semi-blind deconvolution assuming Gaussian blur have introduced this formulation in the functional. The authors of [11] assume that Gaussian blur is the cause of degradation of the image and so replace the kernel term with a Gaussian equation

$$h_\sigma(x,y) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{x^2+y^2}{2\sigma^2}\right) \tag{3.38}$$

which is dependent on the variable σ. The objective functional for the restoration is similar to total variation regularised deblurring [169, 54] made up of a fitting term for deblurring in the presence of Gaussian noise and a smoothness term given as the L_2 norm of the gradient which allows for smooth kernels, in contrast to total variation which permits piecewise constant functions. The equation is modified to allow for the

parametrically defined kernel function h_σ. It is thus presented as

$$f^{BSK}(u,\sigma) = \frac{1}{2}\int_\Omega (h_\sigma * u - z)^2 dA + \gamma \int_\Omega |\nabla h_\sigma|^2 dA$$

where γ is a regularisation parameter controlling the smoothness of the recovered blur function, u is the restored image and z is the received data. In order to minimise this functional, the authors derive the Euler Lagrange equations for u and σ and proceed with alternate minimisation of the arguments.

[64] also considers the case of semi-blind parametric deblurring in a constrained blind deconvolution problem by assuming that the blur function may be of Gaussian type and so modelled as (3.38), leaving only a vector containing a small number of parameters to be found.

The representation of a blur kernel by a differentiable function which allows it to be incorporated into the minimisation problem is useful but in the above work is limited to Gaussian functions which cannot be used to approximate many blur types. In Chapter 7, we consider an extension to this which allows more classes of blur to be incorporated into an energy functional.

3.4 Blind Image Deblurring (BID)

Blind image deconvolution addresses the challenging case of recovering an image from blur degradation without any knowledge or assumptions about the blur kernel or point spread function.

Such models typically involve the simultaneous approximation of both the image and the blur function. There also exist algorithms which aim to identify the blur function by analysing image statistics [17, 30, 75, 76, 92, 123, 131, 168, 191]. The identified blur function may then be used to deblur the image. In this section, we will review several some important simultaneous blind deconvolution techniques.

3.4.1 You and Kaveh (1996)

The authors of [216] propose an energy minimisation model which aims to simultaneously recover both the image and the blur function by minimising a functional composed of a fitting or data fidelity term and regularisation of both the kernel and the image

$$f^{YK}(u,h) = \frac{1}{2}\underbrace{\|h(\boldsymbol{x}) * u(\boldsymbol{x}) - z(\boldsymbol{x})\|^2_{L^2(\Omega)}}_{\text{Data Fitting}} + \underbrace{\lambda L_1(u(\boldsymbol{x})) + \gamma L_2(h(\boldsymbol{x}))}_{\textit{Regularisation}} \tag{3.39}$$

where λ and γ are regularisation parameters which are used to adjust the trade-off between data fitting and regularisation for the image and blur function respectively. The authors used the H^1 semi-norm in order to provide regularisation for the image

and blur functions, given by

$$L_1(u(\boldsymbol{x})) = |u(\boldsymbol{x})|_{H^1} = \|\nabla u(\boldsymbol{x})\|_{L^2(\Omega)}, \qquad L_2(h(\boldsymbol{x})) = |h(\boldsymbol{x})|_{H^1} = \|\nabla h(\boldsymbol{x})\|_{L^2(\Omega)}.$$

This regularisation can be regarded as a smoothness constraint, which encourages the image and blur function to be smooth thereby removing noise. This is a useful constraint for images corrupted Gaussian blur and can provide good results for images with many smooth regions, including those with some sharp edges.

Additionally, the following three constraints are imposed. The first aims to restrict the restored image to non-negative intensity values, that is

$$0 \leq u(\boldsymbol{x}) < \infty \qquad \forall \boldsymbol{x} \in \Omega \tag{3.40}$$

since image intensity is finite and non-negative. Secondly, the sum of the absolute values of that point spread function should be equal to one

$$\sum_{\boldsymbol{x} \in \Omega} h(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x} = 1 \tag{3.41}$$

since an imaging system does not generate or absorb energy and the image intensities are non-negative. Finally, the values of the point spread function should also be non-negative

$$h(\boldsymbol{x}) \geq 0 \qquad \forall \boldsymbol{x} \in \Omega. \tag{3.42}$$

In order to minimise the functional f^{YK} given by (3.39), a strategy of iterative optimisation procedures is used, and the the constraints (3.40), (3.41) and (3.42) are imposed at each iteration. The Euler-Lagrange equations obtained from the gradient of (3.39) are given by

$$\frac{\partial}{\partial u} f^{YK}(u): \;\; h^\dagger(\boldsymbol{x}) * (h(\boldsymbol{x}) * u(\boldsymbol{x}) - z(\boldsymbol{x})) + \lambda \Delta u(\boldsymbol{x}) = 0, \tag{3.43}$$

$$\frac{\partial}{\partial h} f^{YK}(h): \;\; u^\dagger(\boldsymbol{x}) * (u(\boldsymbol{x}) * h(\boldsymbol{x}) - z(\boldsymbol{x})) + \gamma \Delta h(\boldsymbol{x}) = 0. \tag{3.44}$$

The authors propose to solve the deblurring problem by minimising (3.39) with respect to the kernel and the image separately and alternately. Letting

$$f^{YK}(u, h \,|\, u) = f^{YK}(u, h) \text{ where } u \text{ is fixed}, \tag{3.45}$$

$$f^{YK}(u, h \,|\, h) = f^{YK}(u, h) \text{ where } h \text{ is fixed}, \tag{3.46}$$

the algorithm can be given as follows, where R denotes a matrix of random numbers.

1: Make initial estimates of the image and blur function

$$u^{(0)} = z, \qquad h^{(0)} = R$$

2: **for** $\ell \leftarrow 1 : maxit$ **do**

3: Update $u^{(\ell)}$:

$$u^{(\ell)} \leftarrow \min_u f^{YK}\Big(u^{(\ell-1)}, h^{(\ell-1)} \,|\, h^{(\ell-1)}\Big)$$

4: Update $h^{(\ell)}$:

$$h^{(\ell)} \leftarrow \min_h f^{YK}\Big(u^{(\ell-1)}, h^{(\ell-1)} \,|\, u^{(\ell-1)}\Big)$$

5: **end for**

In order to solve equations (3.43) and (3.44) the steepest descent method is considered. While this method can yield good results, the use of the Conjugate Gradient method is preferred due to the increased speed at a cost of only a small increase in complexity.

In order to improve the speed at which a solution may be obtained it is acknowledged that assuming a large support for the point spread function may result in many unnecessary calculations, yielding a computationally expensive procedure. It is therefore proposed to reduce the size of the support for the point spread function defined on a domain $\mathcal{D}$. It must be considered that if the specified support does not actually cover the required support for the true point spread function then the blur function may not be correctly identified.

It is therefore proposed to begin the algorithm with a large support for the point spread function and reduce its size at each iteration assuming that the value of the blur function at the boundary $\delta\mathcal{D}$ is sufficiently small. This is done by removing points in a process known as *pruning*. For convenient implementation, we aim to avoid obtaining an irregular domain by pruning an entire edge $\partial\bar{\mathcal{D}}$ if the point spread values are all sufficiently small on that edge, that is if

$$h(\boldsymbol{x}) \leq \Lambda \qquad \forall h(\boldsymbol{x}) \in \partial\bar{\mathcal{D}}$$

for a given threshold value Λ.

3.4.2 Chan and Wong (1998)

The authors of [54] proposed an improvement to the model by You and Kaveh [216] described above. The aim is to minimise a functional of the type (3.39). The usage of the H^1 semi-norm of the image as a regularisation function is able to obtain good results for reconstructing piecewise smooth functions. However, many images contain large jumps in intensity values which may appear over-smoothed. The case is similar with application to the regularisation of point spread functions for which such a smoothness constrain may not be appropriate. There are many classes of blur function which contain significant jumps in value or which are piecewise constant, such as linear motion or out of focus blur.

Given these considerations, it is proposed to use total variation regularisation for restoring both the blur function and the image due to its ability to obtain good quality results in smooth regions while being able to restore edges. It is therefore proposed to

recover the blur function and the image by finding the minimisation of the functional (3.39) with the functions

$$L_1(u(\boldsymbol{x})) = |u|_{TV} = \int_\Omega |\nabla u(\boldsymbol{x})|, \qquad L_2(h(\boldsymbol{x})) = |h|_{TV} = \int_\Omega |\nabla h(\boldsymbol{x})|.$$

In order to minimise this functional, we replace the total variation function with a smooth approximation, given by

$$|u(\boldsymbol{x})|^\beta_{TV} = \int_\Omega |\nabla u(\boldsymbol{x})|_\beta = \int_\Omega \sqrt{|\nabla u(\boldsymbol{x})|^2 + \beta^2}$$

(similarly for the point spread function h) which can be adjusted by the small parameter $\beta \in \mathbb{R}$. Typically, larger β results a smoother image while lower β leads to image becoming piecewise constant by the *staircasing effect*.

Inclusion of this regularisation function in (3.39) yields the Chan-Wong functional given by

$$f^{CW}(u,h) = \frac{1}{2}\underbrace{\|h(\boldsymbol{x}) * u(\boldsymbol{x}) - z(\boldsymbol{x})\|^2_{L^2(\Omega)}}_{\text{Data Fitting}} + \underbrace{\alpha_1 \int_\Omega |\nabla u(\boldsymbol{x})|_\beta + \alpha_2 \int_\Omega |\nabla h(\boldsymbol{x})|_\beta}_{\textit{Regularisation}} \tag{3.47}$$

where α_1 and α_2 are small, non-negative parameters used to control the amount of regularisation used.

We can observe that $f^{CW}(u, h \,|\, u)$ and $f^{CW}(u, h \,|\, h)$, defined similarly to (3.45) and (3.46), are convex with respect to u and h respectively. It is important to note that, as is the case with (3.39), the functional (3.47) is not jointly convex. In order to address this issue, the functional (3.47) is minimised subject to the constraints similar to (3.40), (3.41) and (3.42) which ensure that the image and blur function are non-negative and that the blur function has a unit integral. Additionally, the point spread function is assumed to be symmetric. The constraints are thus given by

$$u(\boldsymbol{x}) \succeq 0, \qquad h(\boldsymbol{x}) \succeq 0, \qquad \int_\Omega h(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} = 1, \qquad h(\boldsymbol{x}) = h(-\boldsymbol{x}). \tag{3.48}$$

In order to minimise this model, we derive the associated Euler-Lagrange equations given by

$$\frac{\partial}{\partial u} f^{CW}(u,h): \;\; h^\dagger(\boldsymbol{x}) * (h(\boldsymbol{x}) * u(\boldsymbol{x}) - z(\boldsymbol{x})) + \alpha_1 \nabla \cdot \left(\frac{u(\boldsymbol{x})}{|u(\boldsymbol{x})|_\beta}\right) = 0, \tag{3.49}$$

$$\frac{\partial}{\partial h} f^{CW}(u,h): \;\; u^\dagger(\boldsymbol{x}) * (u(\boldsymbol{x}) * h(\boldsymbol{x}) - z(\boldsymbol{x})) + \alpha_2 \nabla \cdot \left(\frac{h(\boldsymbol{x})}{|h(\boldsymbol{x})|_\beta}\right) = 0. \tag{3.50}$$

A scheme of alternate minimisation of the functional (3.47) is developed by alternately solving (3.49) and (3.50) and imposing the constraints (3.48) after each iteration as shown in Algorithm 6.

Algorithm 6 Chan-Wong [54] Method (CW)

Make initial estimates of the image and blur function

$$u^{(0)} \leftarrow z, \qquad h^{(0)} \leftarrow \delta(\boldsymbol{x})$$

function $(h, u) \leftarrow \mathrm{CW}(h^{(0)}, u^{(0)}, z, \alpha_1, \alpha_2, maxit)$

  for $\ell \leftarrow 1 : maxit$ **do**

    Update the image u by solving for $u^{\left(\ell-\frac{1}{2}\right)}$:

$$h^{(\ell-1)}(-\boldsymbol{x}) * \left(h^{(\ell-1)}(\boldsymbol{x}) * u^{\left(\ell-\frac{1}{2}\right)}(\boldsymbol{x}) - z(\boldsymbol{x})\right) + \alpha_1 \nabla \cdot \left(\frac{u^{\left(\ell-\frac{1}{2}\right)}(\boldsymbol{x})}{|u^{\left(\ell-\frac{1}{2}\right)}(\boldsymbol{x})|_\beta}\right) = 0$$

    Impose the non-negativity constraint $u^{(\ell)} \leftarrow \mathbb{P}\left(u^{\left(\ell-\frac{1}{2}\right)}\right)$

    Update the blur function h by solving for $h^{\left(\ell-\frac{1}{4}\right)}$:

$$u^{(\ell)}(-\boldsymbol{x}) * (u^{(\ell)}(\boldsymbol{x}) * h^{\left(\ell-\frac{1}{4}\right)}(\boldsymbol{x}) - z(\boldsymbol{x})) + \alpha_2 \nabla \cdot \left(\frac{h^{\left(\ell-\frac{1}{4}\right)}(\boldsymbol{x})}{|h^{\left(\ell-\frac{1}{4}\right)}(\boldsymbol{x})|_\beta}\right) = 0$$

    Impose the non-negativity constraint $h^{\left(\ell-\frac{2}{4}\right)}(\boldsymbol{x}) \leftarrow \mathbb{P}\left(h^{\left(\ell-\frac{1}{4}\right)}(\boldsymbol{x})\right)$

    Impose the symmetry constraint $h^{\left(\ell-\frac{3}{4}\right)} \leftarrow \left(h^{\left(\ell-\frac{2}{4}\right)}(\boldsymbol{x}) + h^{\left(\ell-\frac{2}{4}\right)}(-\boldsymbol{x})\right)/2$

    Impose the unit integral constraint $h^{(\ell)}(\boldsymbol{x}) \leftarrow h^{\left(\ell-\frac{3}{4}\right)}(\boldsymbol{x}) / \int_\Omega h^{\left(\ell-\frac{3}{4}\right)}(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}$

  end for

  $h \leftarrow h^{(k+1)}$

  $u \leftarrow u^{(k+1)}$

end function

Note that the δ-function and projection $\mathbb{P}$ are respectively defined as

$$\delta(\boldsymbol{x}) = \begin{cases} 1 & \text{if } \boldsymbol{x} = \mathbf{0} \\ 0 & \text{otherwise} \end{cases} \qquad \mathbb{P}(u(x)) = \begin{cases} 0 & \text{if } \leq u(x) < 0 \\ u(x) & \text{if } 0 \leq u(x) \leq 2^{\varphi} \\ 2^{\varphi} & \text{if } \leq 2^{\varphi} < u(x) \end{cases}$$

where the projection is defined point-wise and φ is the bits-per-sample rate of the image.

3.4.3 Perrone and Favaro (2014)

The problem of blind deconvolution was addressed recently in a paper by Perrone and Favaro [156] who aim to simultaneously restore the image and blur function by minimising a functional given some constraints. In order to allow the approximation of a wider range of blur functions, a priori assumptions about the smoothness is not considered since blur functions may be smooth (as in the case of that representing Gaussian blur corruption) or piecewise constant (as with out of focus and linear motion blur). The minimisation problem is thus presented as

$$\min_{u(\boldsymbol{x}),h(\boldsymbol{x})} \left\{ f^{PF}(u(\boldsymbol{x}), h(\boldsymbol{x})) = \int_{\Omega} ([h * u](\boldsymbol{x}) - z(\boldsymbol{x}))\, \mathrm{d}\boldsymbol{x} + \lambda J(u(\boldsymbol{x})) \right\} \tag{3.51}$$

where $J(u(\boldsymbol{x}))$ represents a smoothness constraint on the image $u(\boldsymbol{x})$ in the form of regularisation and $\lambda \in \mathbb{R}_{>0}$ is a small non-negative regularisation parameter. The image regularisation function proposed is given by the total variation semi-norm as

$$J(u(\boldsymbol{x})) = \|\nabla u(\boldsymbol{x})\|_{TV} = \int_{\Omega} |\nabla u(\boldsymbol{x})|$$

where $\nabla u(\boldsymbol{x})$ is the gradient of the function $u(\boldsymbol{x})$. The functional f^{PF} given in equation (3.51) is minimised subject to the non-negativity and unit integral constrains given on the blur function as

$$h(\boldsymbol{x}) \succeq 0, \qquad \|h(\boldsymbol{x})\|_1 = \int_{\Omega} |h(\boldsymbol{x})|\, \mathrm{d}\boldsymbol{x} = 1.$$

The authors propose the implementation of these constraints by projections. In the case of the non-negativity constraint, any negative value of the blur function is set to zero by the projection $\mathbb{P}^{PF}(h(\boldsymbol{x}))$ given in equation (3.52). In order to enforce the unit integral constraint, the blur function is rescaled by the mapping $\mathcal{C}^{PF}(h(\boldsymbol{x}))$ given in equation (3.52).

$$\mathbb{P}^{PF}(h(\boldsymbol{x})) = \begin{cases} h(\boldsymbol{x}) & \text{if } h(\boldsymbol{x}) > 0 \\ 0 & \text{otherwise,} \end{cases} \qquad \mathcal{C}^{PF}(h(\boldsymbol{x})) = \frac{h(\boldsymbol{x})}{\int_{\Omega} h(\boldsymbol{x})\, \mathrm{d}\boldsymbol{x}}. \tag{3.52}$$

The equation (3.51) is minimised using a scheme of alternate minimisation of the arguments. This results in the sub-problems of solving the resulting Euler Lagrange

equations for which gradient descent schemes given by the discretisation of equations (3.53)–(3.54) given below in time and space.

$$\frac{\partial u(t,\boldsymbol{x})}{\partial t} = h^{\dagger}(\boldsymbol{x}) * (h(\boldsymbol{x}) * u(t,\boldsymbol{x})) - \lambda \nabla \cdot \left(\frac{\nabla u(t,\boldsymbol{x})}{|\nabla u(t,\boldsymbol{x})|} \right), \tag{3.53}$$

$$\frac{\partial h(t,\boldsymbol{x})}{\partial t} = u^{\dagger}(\boldsymbol{x}) * (u(\boldsymbol{x}) * h(t,\boldsymbol{x})), \tag{3.54}$$

where $h^{\dagger}(\boldsymbol{x}) = h(-\boldsymbol{x})$ and $u^{\dagger}(\boldsymbol{x}) = u(-\boldsymbol{x})$. The authors attempt to achieve minimisation of the functional using a projected alternate minimisation (pam) which involves alternately moving in one time step for each sub-problem and enforcing the constraints. The newly found blur function is then used in order to obtain the next update of the image and vice versa. This can be seen in the below algorithm. For ease of notation, we use $h_{-} = h^{\dagger}$, and similarly for the image. We use $\odot$ to denote the discrete convolution, time steps ϵ_u and ϵ_h to denote the time steps resulting from the minimisation of equations (3.53) and (3.54) respectively. In order to accommodate sub-integer time steps, we denote by ℓ_b^a the time step $\ell + a/b$. Since the regularisation parameter is difficult to select for a given problem, the algorithm begins with a large parameter λ resulting in more emphasis placed on the smoothness constraint and gradually reduces λ so that the deblurring term plays more of a role.

1: Make initial estimates of the image and blur function

$$u^{(0)} \leftarrow z, \qquad h^{(0)} \leftarrow \delta(\boldsymbol{x})$$

2: **for** $\ell \leftarrow 1 : maxit$ **do**

3: Update the image $u^{\ell+1}$ evaluating:

$$u^{\ell+1} \leftarrow u^{\ell} - \epsilon_u \left(h_{-}^{\ell} \odot \left(h^{\ell} \odot u^{\ell} - z \right) - \lambda \nabla \cdot \frac{\nabla u^{\ell}}{|\nabla u^{\ell}|} \right)$$

4: Update the blur function $h^{\ell_3^1}$ evaluating:

$$h^{\ell_3^1} \leftarrow h^{\ell} - \epsilon_h \left(u_{-}^{\ell+1} \odot \left(u^{\ell+1} \odot h^{\ell} - z \right) \right)$$

5: Impose the non-negativity constraint $h^{\ell_3^2} \leftarrow \mathbb{P}^{PF}\left(h^{\ell_3^1}\right)$

6: Impose the unit integral constraint $h^{\ell+1} \leftarrow \mathcal{C}^{PF}\left(h^{\ell_3^2}\right)$

7: $\lambda \leftarrow \max\{0.99\lambda, \lambda_{min}\}$

8: **end for**

9: $u \leftarrow u^{\ell+1}$

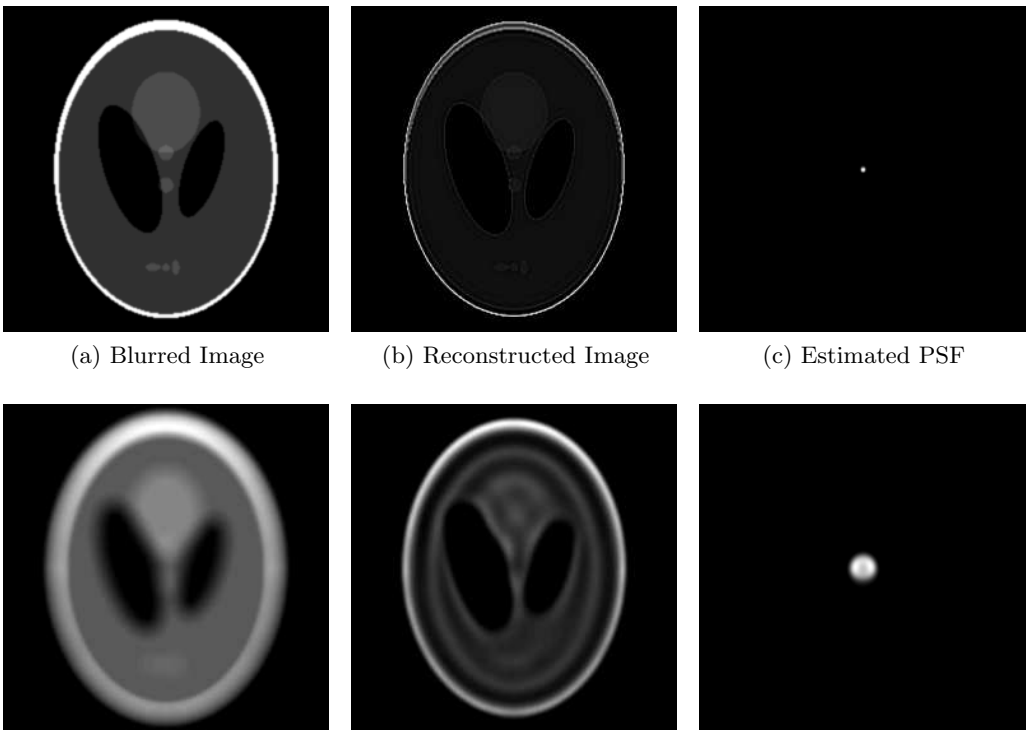
10: $h \leftarrow h^{\ell+1}$

3.4.4 Matlab Deblurring

The Matlab software contains a code which may be used to attempt to deblur an image and recover its point spread function. It may be implemented with the command

```
[J, PSF] = deconvblind(I, INITPSF)
```

where `I` is a blurred image and `INITPSF` is an initial estimate of the point spread function. This command can obtain reasonable results for small amounts of blur but many defects are introduced into the image, which are more evident with stronger levels of blur. See Figure 3.8.



(a) Blurred Image (b) Reconstructed Image (c) Estimated PSF

(d) Blurred Image (e) Reconstructed Image (f) Estimated PSF

Figure 3.8: Illustration of the performance of Matlab's `deconvblind` command. On the top row, the "phantom" image of size 256×256 has been blurred with an out-of-focus blur function of diameter 3. On the bottom row, the same image has been blurred by an out-of-focus blur function of diameter 21. On both rows, from left to right we have (a, d) the blurred image, (b, e) the restored image and (c, f) the estimated blur function. In both cases, the blur function is approximately but not accurately estimated and the restored images contain many defects which are more obvious in the 2nd example (e).

3.4.5 Deblurring of Multi-Channel Images

Multi-channel or colour images may be represented as a $m \times n \times p$ matrix where $p = 3$ for RGB images. The above deblurring models may be applied to such images by converting the image to a grayscale counterpart by a weighted sum of the channels

$$u^{gray}(x, y) = \gamma_R u^{red}(x, y) + \gamma_G u^{green}(x, y) + \gamma_B u^{blue}(x, y)$$

where the coefficients are chosen to attempt to approximate the human perception of luminance. As an example, the relative luminance can be calculated with

$$(\gamma_R, \gamma_G, \gamma_B) = (0.2126, 0.7152, 0.0722).$$

The grayscale image may then be deblurred to reveal a sharp image, but this results in a grayscale image which can be useful for testing models without additional computation for individual channels or for images received in grayscale but may not be desirable for real applications to colour images.

To address this issue, we may deblur each channel of the image individually in order to obtain the restored images $\bar{u}^{red}$, $\bar{u}^{green}$ and $\bar{u}^{blue}$. Once these approximations are obtained, the colour image can be reconstructed by reformulating the $3D$ matrix. This enables us to obtain a sharp colour image for a blurred image.

Treating each channel of an image independently can allow us to obtain a good approximation of the true image but can result in misaligned edges between channels which leads to undesirable effects in the final restored image. In the case of blind deblurring, the point spread function may not be correctly identified in each channel leading to a different approximation of the image. For example, colour Fundus images typically contain few details in the blue channel resulting in a very smooth image in contrast to the green channel which contains much information. Efforts have therefore been made to include multichannel information in the restoration of the image.

Deblurring in Colour Space

Kaftory, Sochen and Zeevi [112] attempt to use inter-channel information by minimising the Polyakov action. This causes the image to become smoother and the individual channels to align which helps edges to be better reconstructed. [114, 183] The Polyakov action is adopted as the regularisation function so that the problem is formulated as

$$\min_{u^a,h} \left\{ f^{KSZ}(u^a, h) = \frac{1}{2} \sum_a \int_\Omega ([h * u](\boldsymbol{x}) - z(\boldsymbol{x}))^2 \, \mathrm{d}\boldsymbol{x} + \alpha_1 S\left(u^a(\boldsymbol{x})\right) + \alpha_2 S(h(\boldsymbol{x})) \right\}. \tag{3.55}$$

where the function S may be given as

$$S\left(u^a\right) = \int_\Omega \sqrt{1 + \beta^2 \sum_a |\nabla u^a|^2 + \frac{1}{2}\beta^4 \sum_{a,b} \left(\nabla u^a, \nabla u^b\right)^2} \, \mathrm{d}\boldsymbol{x}$$

for the image u, where $\left(\nabla u^a, \nabla u^b\right)$ is used to denote the magnitude of the vector product, and

$$S(h) = \int_\Omega \sqrt{1 + \beta^2 |\nabla h|^2} \, \mathrm{d}\boldsymbol{x}.$$

An alternate minimisation scheme is used to recover the image channels u^a and the point spread function h. Some examples of deblurring multichannel images are shown in Figure 3.9.

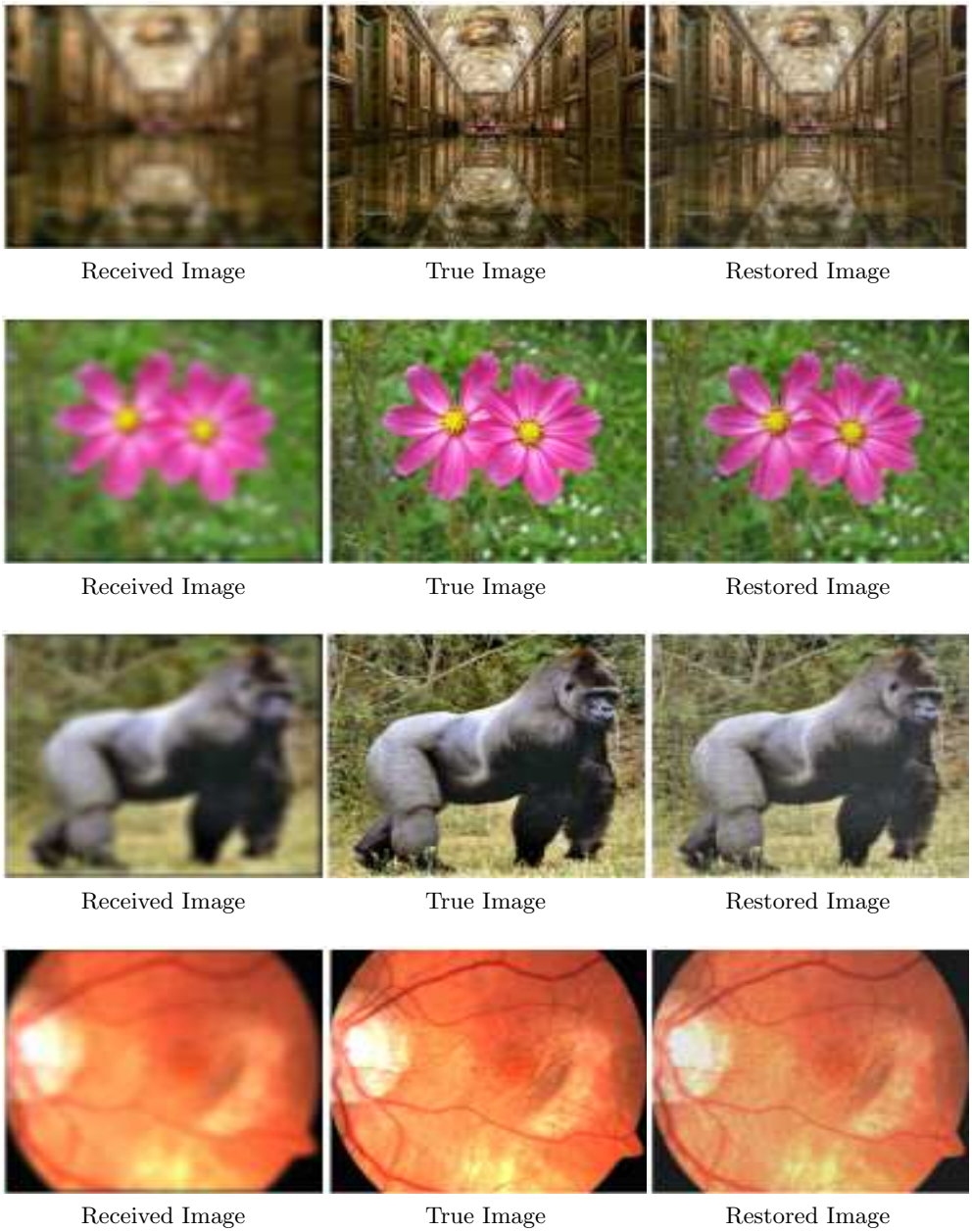
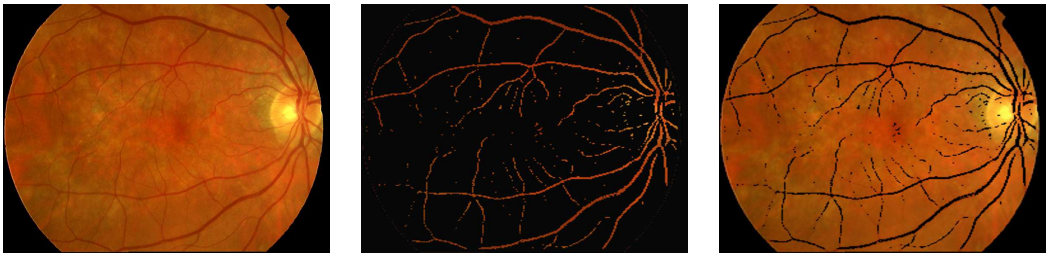


Figure 3.9: Illustration of restoring multichannel images from blur corruption using inter-channel information by solving the minimisation problem (3.55) with known point spread function. From top to bottom, we have the examples 1) Apollo Gallery, 2) Aster, 3) Gorilla, 4) Colour Fundus Retina. On each row, from left to right, we have 1) the received data, 2) the true image which we want to approximate, 3) the restored image. In each case, the image is successfully restored and edges can be clearly seen.

3.5 Image Segmentation

Image segmentation is a significant problem in image processing which aims to separate the foreground and background of an image in order to select certain features. See Figure 3.10 for some examples. It has many applications, including usage in medical imaging where the aim may be to select a particular organ and CCTV monitoring of a subject.

In this section, we briefly review the Mumford-Shah image segmentation model [143] which aims to give a piecewise smooth approximation to an image and the Chan-Vese model [53] aims to find a piecewise constant approximation of the true image where certain features are represented by a constant intensity. These models will be useful to know for Chapter 8.



(a) Colour Fundus Retina Image

(b) Extracted Blood Vessels

(c) Image Minus Blood Vessels

(d) Tree Image

(e) Segmentation of Tree Image

(f) Binary Representation

Figure 3.10: Examples to demonstrate image segmentation. On the top row, we show retinal vessel segmentation of a Colour Fundus Angiography image. From left to right, we have the (a) image to be segmented, (b) the extracted blood vessels and (c) the image without the vessels which have been extracted. On the bottom row, we show the segmentation of the tree image (d). In image (e) we see that the tree and landscape can be separated from the sky by the red lines. Image (f) shows the binary representation of this segmentation, where white means foreground and black means background.

3.5.1 Mumford-Shah Segmentation Model

Mumford and Shah [143] introduced the problem of attempting to approximate a given image by a piecewise smooth function. They present this as a minimisation problem

given by

$$\min_{\Gamma,c_1,c_2} \Big\{ f^{MS}(\Gamma(\boldsymbol{x}), c_1, c_2) = \mu \text{length}\left(\Gamma(\boldsymbol{x})\right)$$

$$+\lambda_1 \int_{\text{inside}(\Gamma)} |z(\boldsymbol{x}) - c_1|^2 \, \mathrm{d}\boldsymbol{x} + \lambda_2 \int_{\text{outside}(\Gamma)} |z(\boldsymbol{x}) - c_2|^2 \, \mathrm{d}\boldsymbol{x} \Big\} \tag{3.56}$$

where Γ is the contour of the segmentation, c_1 is the average of the intensities of the pixels inside the shape defined by the contour Γ and c_2 is the average of the intensities of the pixels outside of the shape defined by Γ. μ, λ_1 and λ_2 are positive parameters. The first term of (3.56) aims to minimise the length of the contour Γ. The aim of the second and third terms are to drive c_1 and c_2 close to the average intensities of the inside and outside respectively of the shape defined by Γ.

3.5.2 Chan-Vese Segmentation Model

Chan and Vese [53] applied the level set method, introduced by Osher and Sethian [151], to solve the binary labelling problem above (3.56), and has been very successful in segmenting the features of an image and widely used for developing new models [8, 162, 163, 51]. By representing Γ as the zero level set of the Lipschitz function $\phi : \Omega \to \mathbb{R}$ and introducing the Heaviside step function, $H(x)$, the problem was reformulated as

$$\min_{\phi,c_1,c_2} \Big\{ f^{CV}(\phi(\boldsymbol{x}), c_1, c_2) = \mu \int_\Omega |\nabla H(\phi(\boldsymbol{x}))|$$

$$+\lambda_1 \int_\Omega |z(\boldsymbol{x}) - c_1|^2 H(\phi(\boldsymbol{x})) \, \mathrm{d}\boldsymbol{x} + \lambda_2 \int_\Omega |z(\boldsymbol{x}) - c_2|^2 (1 - H(\phi(\boldsymbol{x}))) \, \mathrm{d}\boldsymbol{x} \Big\} \tag{3.57}$$

which allowed a solution to (3.56) to be found.

Note that the segmentation contour Γ is given by the non-zero elements of the delta function $\delta(\phi)$ evaluated over ϕ, which is assumed to be positive inside the shape defined by Γ (corresponding to $H(\phi) = 1$) and negative outside of the shape (corresponding to $H(\phi) = 0$). In order to minimise this, we calculate the first order optimality conditions given by the derivatives of f^{CV} with respect to the arguments. To do this, we replace the Heaviside and delta functions with analytic approximations $H_\varepsilon(x)$ and $\delta_\varepsilon(x) = H'_\varepsilon(x)$, with

$$H_\varepsilon(x) = \frac{1}{2} + \frac{1}{\pi} \arctan\left(\frac{x}{\varepsilon}\right),$$

where $\varepsilon > 0$ is a small parameter and the approximation tends to the Heaviside function as ε tends to zero. We may then derive the equations to give c_1 and c_2. Then, keeping these fixed, we derive the partial differential equation (pde) for ϕ which can be solved using time marching.

The segmentation models described above are able to achieve good results with clean images, even in the presence of noise. They begin to struggle to capture object boundaries exactly or to identify objects in the presence of blur. We consider a way to overcome this in Chapter 8.

Chapter 4

Application to Blurred Images and some Refinements

We have considered some relevant preliminary mathematics in Chapter 2 and reviewed some relevant models in image deblurring in Chapter 3 along with the introduction of some work in image denoising and image segmentation. In this chapter, we consider the application of some image deblurring ideas discussed in the previous chapter to blurred images. We also present in §4.1 some modifications and refinements of these models by considering the boundary conditions and the solution speed. We next present an accelerated model in §4.2 and an alternative kernel filtering model in §4.3 which helps to deal with small amounts of noise in the restored blur functions.

4.1 Initial Applications

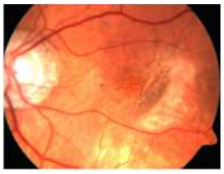
In this section, we present an approach which aims to alleviate the problem of blur in images by implementing the total variation deblurring strategy introduced in §3.3.5. Following this, we attempt to deblur images without knowledge of the blur function by simultaneously restoring the image and point spread functions, following §3.4.2. We then attempt to obtain improvement results by modifying the boundary conditions.

4.1.1 Blur in Medical Images

Blurring of images is a common problem which occurs in many areas. Retinal imaging in particular is an area in which blurring causes significant problems, since blur is frequently present in retinal images and accurate scans are required for diagnosis and screening.

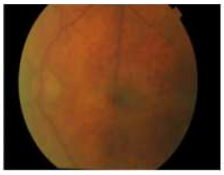
As such, blurring is a major cause for image quality degradation in clinical settings. There are many contributing factors, including eye movement, refractive error and coexisting conditions such as Parkinson's disease. Additional causes include advanced ocular diseases, age and difficulty maintaining careful focus. There exists an established programme for which 3 million diabetic patients undergo photographic screening each year. Approximately 10% or 300,000 of the images are too blurred for assessment.

It would therefore be beneficial to develop deblurring techniques to provide ophthalmologists with tools to enhance the quality of Colour Fundus Autofluorescence retinal images which may increase confidence in diagnosis and treatment planning, reduce the number of investigations required, and ease the development of new therapies. This can be applicable to patients with Diabetic Retinopathy (DR) and Age-related Macular Degeneration (AMD). The findings may also be generalised to other ocular images used in the detection of eye disease, for example Intravenous Fluorescein Angiography (IVFA) as well as non-ophthalmological settings such as Computed Tomography (CT) and Magnetic Resonance Imaging (MRI).

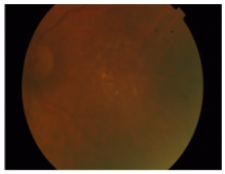


Excellent

Good

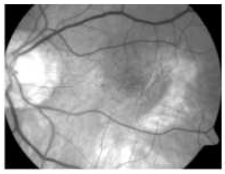


Fair

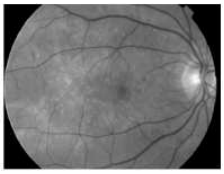


Inadequate

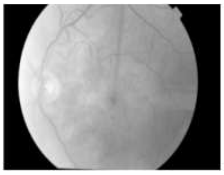
Figure 4.1: Real examples of image quality of Colour Fundus Images.



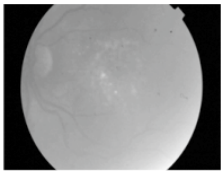
Excellent



Good



Fair



Inadequate

Figure 4.2: Real examples of image quality of Colour Fundus Images (Ch2).

4.1.2 Image Deblurring given the Blur Function

Deblurring as a mathematical consideration is an inverse problem. That is, we wish to solve the equation

$$[\kappa * u](x, y) = z(x, y)$$

for $u(x, y)$. This leads to solving the large system

$$A\boldsymbol{u} = \boldsymbol{b} \qquad \text{where} \qquad \boldsymbol{b} = K^\top \boldsymbol{z} \qquad \text{and} \qquad A = K^\top K + \alpha L,$$

where K is an ill-conditioned (typically full) matrix, z is the received corrupted data, L is a sparse regularisation matrix which may or may not be symmetric, and α is known as the regularisation parameter and is usually small and positive.

In the case of image deblurring, κ is a two-dimensional convolution operator, such that our model is given by

$$[\kappa * u](x, y) = \int\int k(x - x', y - y')u(x', y')\,\mathrm{d}x'\,\mathrm{d}y',$$

where the kernel function k is known as the *point spread function*. With the additional

consideration of additive noise, our model for the observed image is $z(x, y)$ where

$$z(x, y) = [\kappa * u_{true}](x, y) + \eta(x, y)$$

where u_{true} is the true image and η is the additive noise term. Our goal is to estimate the true image u_{true} given the received image z and the kernel function κ without knowledge of the noise term η.

One issue which we must address is that this type of problem is ill-posed, i.e. the estimated image u is very sensitive to small changes in the received data z. To address this issue, we use the idea of Tikhonov regularisation [193] whereby we attempt to solve a penalised least squares problem, given by

$$\min_u \left\{ \|[\kappa * u](x, y) - z(x, y)\|^2_{L^2(\Omega)} + \alpha L(u(x, y)) \right\}$$

where $L(u)$ is a penalty term which aims to reduce the presence of defects arising from the ill-posedness of the problem.

Following §3.3.5, This paper we use the Total Variation regularisation function given by

$$L(u(x, y)) = \int\int |\nabla(u(x, y))|$$

since it is particularly effective for restoring edges in images. The resulting method, proposed by Rudin, Osher and Fatemi [169], is known as the Total Variation (tv) minimisation method [196]. The discretisation of the resulting Euler-Lagrange equation of this method gives us

$$K^\top K\boldsymbol{u} - \alpha\nabla \cdot \left(\frac{\nabla \boldsymbol{u}}{|\nabla \boldsymbol{u}|_\beta} \right) = K^\top \boldsymbol{z}, \tag{4.1}$$

where the discrete image represented by an $m \times n$ matrix u has been transformed to a mn-vector $\boldsymbol{u}$ by the mapping

$$\mathbb{M}(u) \to \boldsymbol{u} \qquad \text{where} \qquad \mathbb{M}\left(u_{i,j}\right) = \boldsymbol{u}_{i+m(j-1)}$$

and K is a dense $mn \times mn$ matrix which represents the blur function such that $\mathbb{M}^{-1}(K\boldsymbol{u})$ gives the discretisation of the convolution $[\kappa * u](x, y)$. In order to avoid division by zero when $|\nabla u| = 0$, we have introduced a small positive parameter β so that our denominator $|\nabla u|$ is replaced by $|\nabla u|_\beta = \sqrt{|\nabla u|^2 + \beta}$. Care should be taken when selecting the value of β. Values which are too small can make restored images look "blocky" whereas values which are too small can give overly smooth results.

Size of the system

One important consideration is the size of the system since K in particular can be extremely large and dense, in contrast with finite difference operators which typically may be represented by large but sparse matrices. For example, a 256×256 pixel image

may be considered as a sensible size to use however, this would yield a $256^2 \times 256^2$ matrix K containing over 4×10^9 entries. The formation of the blur matrix K would limit many computers to only 64×64 pixel images.

Fast Fourier transform

Using the convolution theorem given in §2.8, we may calculate the convolution using the fast Fourier transform (fft) as

$$[\kappa * u](x, y) = \mathcal{F}^{-1}\left(\mathcal{F}(\kappa(x, y))\mathcal{F}(u(x, y))\right).$$

We can then calculate the discrete convolution using the discrete Fourier transform (dft) using matrices of size $2n \times 2n$. For a discrete image of size $m = n = 256$, this results in a system approximately of 2.6×10^5 entries and thus allows us to consider larger images.

4.1.3 Solving the System and Results

Since it is not computationally feasible to solve the system directly, we make use of iterative solvers. Since, with Dirichlet boundary conditions, we can obtain a system $A\boldsymbol{u} = \boldsymbol{b}$ with A symmetric positive definite, the conjugate gradient (CG) method is suitable to find a solution to the system. To accelerate convergence, we make use of preconditioners. One example used is the Product Preconditioner, proposed by Vogel and Oman [195], which is given by

$$P = (1/\gamma)(\tilde{K}\tilde{K} + \gamma)^{1/2}(\gamma I + \alpha L(u))(\tilde{K}\tilde{K} + \gamma)^{1/2} \tag{4.2}$$

where γ is a positive parameter and $\tilde{K}$ is a circulant approximation to the matrix K, such as the Strang approximation [60]. One drawback of using Strang's circulant approximation to a Toeplitz matrix is that much of the information about the kernel can be lost in the preconditioner. Since the preconditioner should be a 'good' approximation to A, the choice of a good circulant approximation is significant. We then solve the system using an adapted preconditioned conjugate gradient (PCG) method.

Experimental results

Results using this method are given in Table 4.1 and Figure 4.3. We test the model by restoring retina and satellite images which have been corrupted by motion and Gaussian blur. We measure the error in the restored image compared with the true image using the signal-to-noise ratio (SNR) given as

$$SNR = 10 \log_{10}\left(\frac{\sum_{x,y} |u_{\text{true}}(x, y)|^2}{\sum_{x,y} |u_{\text{true}}(x, y) - u(x, y)|^2}\right),$$

Algorithm 7 TV Deblurring with Dirichlet Boundary Conditions (TVDD)

function $\boldsymbol{u} \leftarrow$TVDD$(K, \boldsymbol{z}, \alpha, \beta, maxit)$
  $\boldsymbol{u}^{(0)} \leftarrow \boldsymbol{z}$
  Solve (4.1):
  for $i = 1..maxit$ **do**
    Using preconditioned conjugate gradient with (4.2), solve for $\boldsymbol{u}^{(i)}$:

$$K^\top K\boldsymbol{u}^{(i)} - \alpha\nabla \cdot \left(\frac{\nabla \boldsymbol{u}^{(i)}}{|\nabla \boldsymbol{u}^{(i-1)}|_\beta}\right) = K^\top \boldsymbol{z}$$

  end for
  $\boldsymbol{u} \leftarrow \boldsymbol{u}^{(i)}$
end function

where u is the restored image and u_{true} is the true image, and using the peak signal-to-noise ratio (PSNR) given by

$$PSNR = 20\log_{10}\left(\frac{\max_{x,y}|u_{\text{true}}(x,y)|}{\sqrt{\frac{1}{mn}\sum_{x,y}(u_{\text{true}}(x,y) - u(x,y))^2}}\right).$$

Experiment	Received Image		Restored Image	
	SNR	PSNR	SNR	PSNR
Retina image corrupted by motion blur	18.87	24.76	28.67	34.91
Satellite image corrupted by motion blur	8.95	22.74	35.86	49.16
Retina image corrupted by Gaussian blur	11.09	15.63	21.12	27.34
Satellite image corrupted by Gaussian blur	1.58	9.74	25.84	39.45

Table 4.1: Table of SNR and PSNR error values of images restored by Algorithm 7. The received retina and satellite images had been corrupted by motion and Gaussian blur. In each case, the algorithm is able to obtain an improved result.

4.1.4 Alternative Boundary Conditions

Boundary conditions play an important role in our considerations in both the blind and non-blind cases. They affect not only the model one might use to solve the equation, but the way in which the equation may be solved. For example, a matrix may be symmetric under Dirichlet boundary conditions but asymmetric under Neumann boundary conditions.

Recall that, for conjugate gradient, we require that $A = K^\top K + \alpha L(\boldsymbol{u})$ is symmetric. We have that $K^\top K$ is symmetric. If we are using Dirichlet boundary conditions, then our problem is simpler because the tv operator $L(\boldsymbol{u})$ is also symmetric. However, Dirichlet boundary conditions assume that area outside of the images is black. We attempt to improve the quality of the image restoration using Neumann boundary conditions. In this case, we do not have symmetry in the tv operator. We overcome

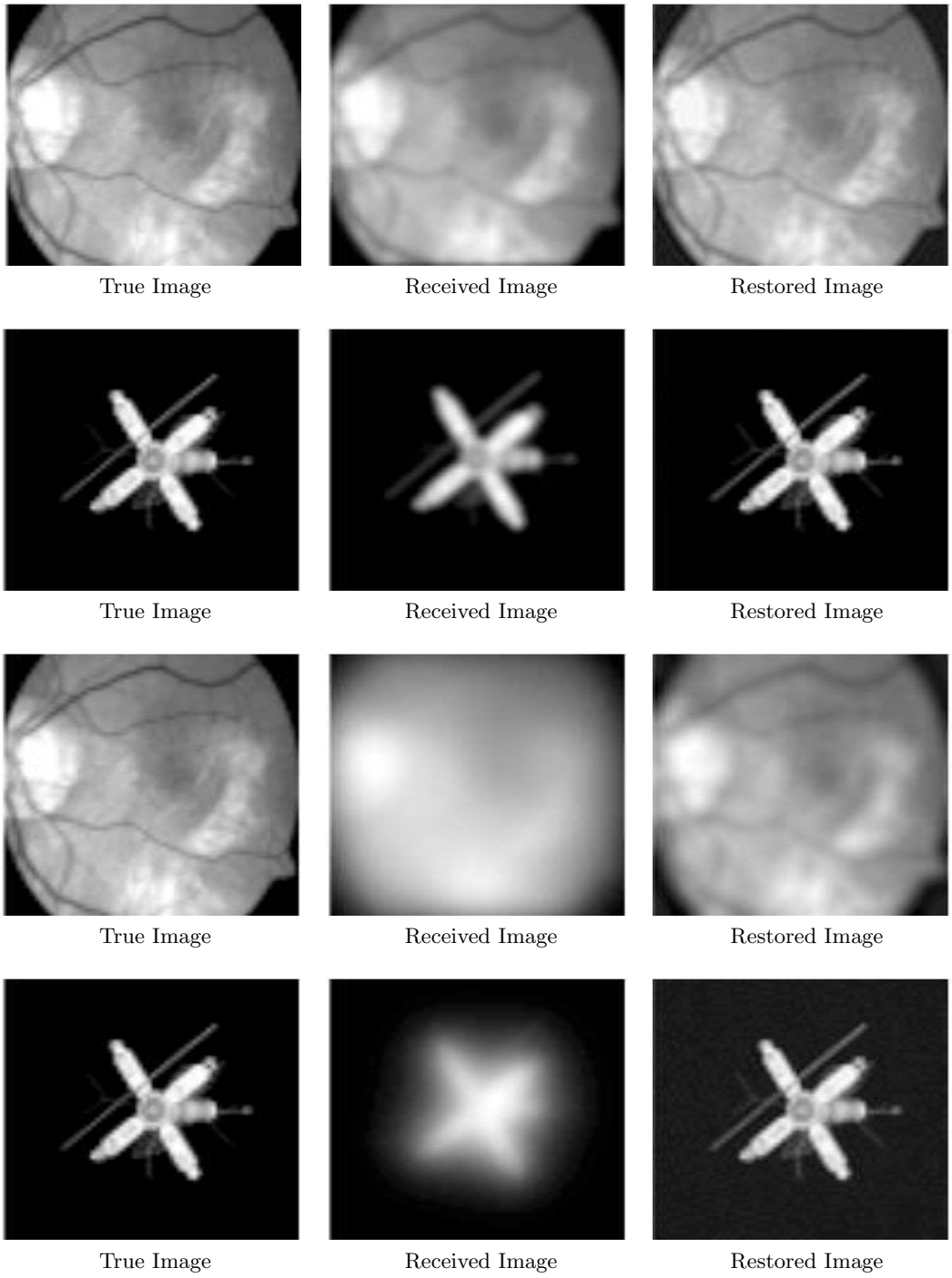


Figure 4.3: Illustration of the deblurring of images by the total variation model given Algorithm 7 assuming Dirichlet boundary conditions. From top to bottom, we have on each row examples of deblurring 1) a Fundus Autofluorescence retina image corrupted by motion blur, 2) a satellite image corrupted by motion blur, 3) a Fundus Autofluorescence retina image corrupted by Gaussian blur, and 4) a satellite image corrupted by Gaussian blur. On each row, from left to right, we have (1) the original (true) image, (2) the received (blurred) image, and (3) the restored (deblurred) image.

this problem by calculating a symmetric approximation L_S^+ and a remainder term L_S^- given by

$$L_S^{\pm}(\boldsymbol{u}) = \frac{L(\boldsymbol{u}) \pm (L(\boldsymbol{u}))^\top}{2}$$

such that $L(\boldsymbol{u}) = L_S^+(\boldsymbol{u}) + L_S^-(\boldsymbol{u})$. We then aim to solve $\hat{A}\boldsymbol{u} = \hat{\boldsymbol{b}}$ where $\hat{A} = K^\top K + \alpha L_S^+(\boldsymbol{u})$ is symmetric and $\hat{\boldsymbol{b}} = K^\top \boldsymbol{z} - \alpha L_S^-(\boldsymbol{u})$, and use a lagging technique to approximate the original problem $A\boldsymbol{u} = \boldsymbol{b}$. To solve for the image $\boldsymbol{u}$, we solve

$$(K^\top K + \alpha L_S^+(\boldsymbol{u}^{(i)})\boldsymbol{u}^{(i)} = K^\top \boldsymbol{z} - \alpha L_S^-(\boldsymbol{u}^{(i-1)})\boldsymbol{u}^{(i-1)},$$

where we iterate on i as shown in Algorithm 8.

Algorithm 8 TV Deblurring with Neumann Boundary Conditions (TVDN)

function $\boldsymbol{u} \leftarrow$TVDN$(K, \boldsymbol{z}, \alpha, \beta, maxit)$
  $\boldsymbol{u}^{(0)} \leftarrow \boldsymbol{z}$
  Solve (4.1):
  for $i = 1..maxit$ **do**
    for $j = 1..maxit$ **do**
      Using preconditioned conjugate gradient with (4.2), solve for $\boldsymbol{u}^{(i,j)}$:

$$(K^\top K + \alpha L_S^+(\boldsymbol{u}^{(i,j-1)})\boldsymbol{u}^{(i,j)} = K^\top \boldsymbol{z} - \alpha L_S^-(\boldsymbol{u}^{(i-1)})\boldsymbol{u}^{(i-1)},$$

    end for
  end for
  $\boldsymbol{u} \leftarrow \boldsymbol{u}^{(i)}$
end function

Preconditioner for conjugate gradient

We now consider appropriate preconditioning for the conjugate gradient algorithm employed in the solution of this problem. Since the discrete Laplacian on a unit square with Neumann boundary conditions can be diagonalised by the discrete cosine transform, Chan and Wong [42] proposed a cosine transform-based preconditioner given by

$$M_D = (C_n \otimes C_n)(\Lambda_1 * \Lambda_1\Lambda_2 * \Lambda_2 + \alpha\Lambda_3)(C_n \otimes C_n)^\top$$

where $\Lambda_1, \Lambda_2, \Lambda_3$ are the eigenvalue matrices of $c_2(K)$, $c_2(\Delta^{-1/2})$ and $c_2(\tilde{L}_{u^k})$ respectively, $\Delta \equiv \rho(K^\top K)I_n + \alpha\text{diag}(L_{u^k})$, $\rho(K^\top K)$ is the spectral radius of $K^\top K$, $\tilde{L}_{u^k} = \Delta^{-1/2} L_{u^k} \Delta^{-1/2}$.

Experimental results

We test the model by restoring retina and satellite images which have been corrupted by motion and Gaussian blur. Results obtained using this method are given in Table 4.2 and Figure 4.4 and compared with the results obtained by implementing Algorithm 7. It

can be noted that the restoration of the retina image is improved by assuming Neumann boundary conditions in the cases of corruption by both motion and Gaussian blur.

Experiment	Received Image		Restored Image (Dirichlet)		Restored Image (Neumann)	
	SNR	PSNR	SNR	PSNR	SNR	PSNR
Retina image corrupted by motion blur	18.87	24.76	28.67	34.91	29.32	35.62
Satellite image corrupted by motion blur	8.95	22.74	35.86	49.16	14.49	28.80
Retina image corrupted by Gaussian blur	11.09	15.63	21.12	27.34	25.23	31.46
Satellite image corrupted by Gaussian blur	1.58	9.74	25.84	39.45	12.98	27.40

Table 4.2: Table of SNR and PSNR error values of images restored by Algorithms 7 and 8. The received retina and satellite images had been corrupted by motion and Gaussian blur. In each case, both algorithms are able to obtain improved results. In the case of the retina image, the case of Neumann boundary conditions provides an improved result.

4.1.5 Image Deblurring without Knowledge of the Blur Function

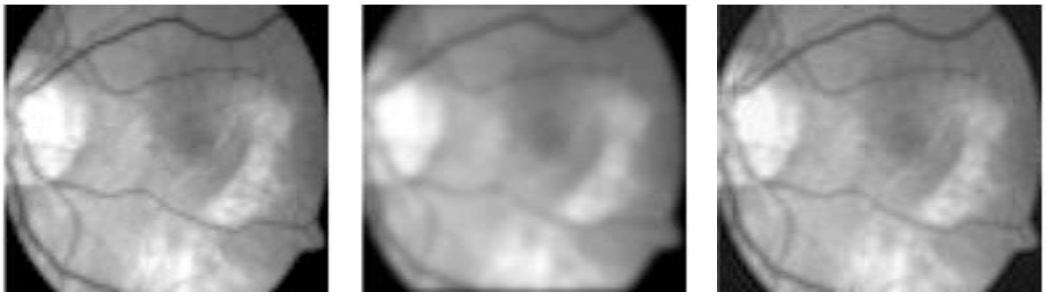
The task of restoring an image from blur corruption without knowledge of the blur function, known as *blind image deconvolution*, is a more challenging yet realistic problem. In this section, we present a method for carrying out blind deconvolution of images by simultaneous approximation of the image and blur functions following §3.4.2 and incorporating alternative boundary conditions.

There exist many algorithms for recovering both the image and the kernel simultaneously. For example, see [78, 121, 120, 216]. Chan and Wong [54] proposed the following alternate minimisation blind deblurring model using total variation:

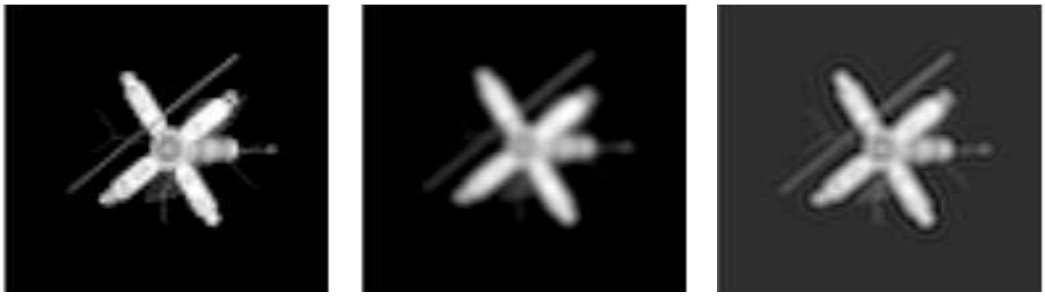
$$\min_{u,h} f(u,h) \equiv \min_{u,h} \frac{1}{2} \left\| [h * u](\boldsymbol{x}) - z(\boldsymbol{x}) \right\|^2_{L^2(\Omega)} + \alpha_1 \int_\Omega |\nabla u(\boldsymbol{x})| + \alpha_2 \int_\Omega |\nabla h(\boldsymbol{x})| \quad (4.3)$$

to recover the image with no a priori information about the point spread function. Here $\boldsymbol{x} = (x, y)$, $z(\boldsymbol{x})$ is the received image, $h(\boldsymbol{x})$ and $u(\boldsymbol{x})$ are the blur function and image respectfully, and α_1, α_2 are small, positive regularisation parameters. Since the total variation method is particularly effective in recovering edges, we present examples of images which have been corrupted by out of focus blur, which is represented by a piecewise constant function. We will attempt to recover both the image and the blur function simultaneously by minimising (4.3).

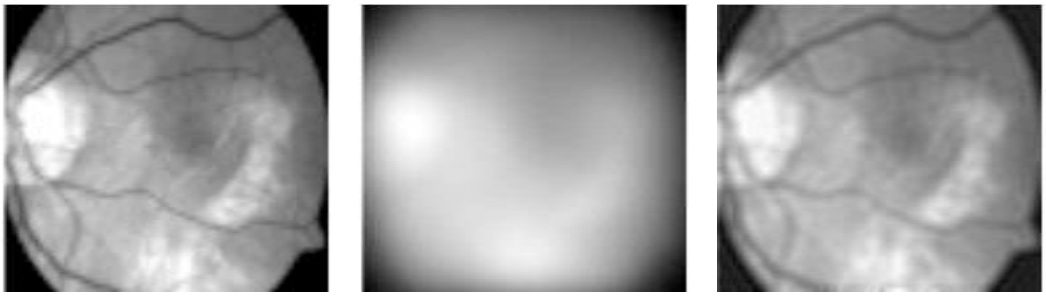
In order to devise a numerical scheme, the following first order optimality conditions



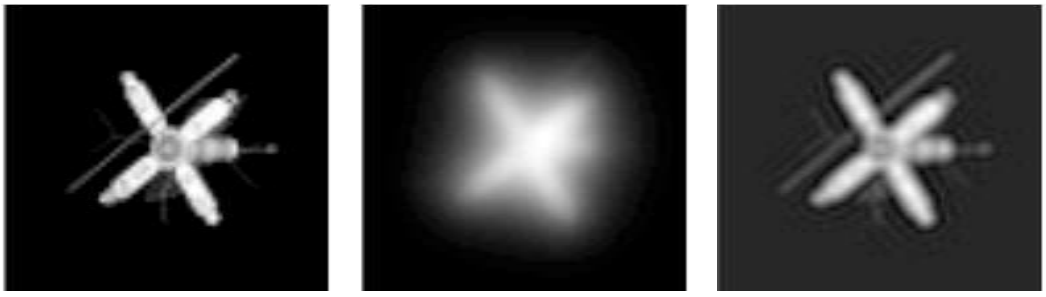
True Image | Received Image | Restored Image



True Image | Received Image | Restored Image



True Image | Received Image | Restored Image



True Image | Received Image | Restored Image

Figure 4.4: Illustration of the deblurring of images by the total variation model given Algorithm 8 assuming Neumann boundary conditions. From top to bottom, we have on each row examples of deblurring 1) a Fundus Autofluorescence retina image corrupted by motion blur, 2) a satellite image corrupted by motion blur, 3) a Fundus Autofluorescence retina image corrupted by Gaussian blur, and 4) a satellite image corrupted by Gaussian blur. On each row, from left to right, we have (1) the original (true) image, (2) the received (blurred) image, and (3) the restored (deblurred) image.

are considered:

$$\frac{\partial f}{\partial h} : u(-\boldsymbol{x}) * (u(\boldsymbol{x}) * h(\boldsymbol{x}) - z(\boldsymbol{x})) - \alpha_2 \nabla \cdot \left(\frac{\nabla h(\boldsymbol{x})}{|\nabla h(\boldsymbol{x})|_\beta} \right) = 0, \qquad \boldsymbol{x} \in \Omega, \tag{4.4}$$

$$\frac{\partial f}{\partial u} : h(-\boldsymbol{x}) * (h(\boldsymbol{x}) * u(\boldsymbol{x}) - z(\boldsymbol{x})) - \alpha_1 \nabla \cdot \left(\frac{\nabla u(\boldsymbol{x})}{|\nabla u(\boldsymbol{x})|_\beta} \right) = 0, \qquad \boldsymbol{x} \in \Omega. \tag{4.5}$$

The idea to solve the blind deblurring model is to develop an alternating minimisation algorithm such that $f\left(u^{(i)}, h^{(i)}\right)$ decreases as i increases.

To solve this system, we first make an initial estimate (u^0, h^0) for the image and blur function (u, h). For the image, it is typical to let the received data provide the initial estimate since it is often the closest approximation available. We then minimise (4.3) by solving $f\left(u^{(0)}, h^{(1)}\right) \equiv \min_h f\left(u^{(0)}, h\right)$ using (4.4), then $f\left(u^{(1)}, h^{(1)}\right) \equiv \min_u f\left(u, h^{(1)}\right)$ using (4.5). At the general step, given $u^{(i)}$ and $h^{(i)}$, we solve

$$\begin{cases} u^{(i)}(-\boldsymbol{x}) * (u^{(i)}(\boldsymbol{x}) * h^{(i+1)}(\boldsymbol{x}) - z(\boldsymbol{x})) - \alpha_2 \nabla \cdot \left(\frac{\nabla h^{(i+1)}(\boldsymbol{x})}{|\nabla h^{(i+1)}(\boldsymbol{x})|_\beta} \right) = 0, & \text{for } h^{(i+1)}(\boldsymbol{x}) \\ h^{(i+1)}(-\boldsymbol{x}) * (h^{(i+1)}(\boldsymbol{x}) * u^{(i+1)}(\boldsymbol{x}) - z(\boldsymbol{x})) - \alpha_1 \nabla \cdot \left(\frac{\nabla u^{(i+1)}(\boldsymbol{x})}{|\nabla u^{(i+1)}(\boldsymbol{x})|_\beta} \right) = 0 & \text{for } u^{(i+1)}(\boldsymbol{x}). \end{cases}$$

One problem we have is that this does not necessarily yield physical solutions because the minimisation problem for (4.3) may not have a unique solution. For example, if (u, h) is a solution, then so are

$$((a_2/a_1)h, (a_1/a_2)u), \qquad (-u, -h), \qquad (u(x \pm c, y \pm d, h(x \mp c, y \mp d)) \, \forall c, d \in \mathbb{R}$$

In order to obtain a physical solution, we impose the following conditions on the image u and the blur function h which ensure that the integral of the blur function is equal to one, both the image and blur function are non-negative, and the blur function is rotationally symmetric:

$$\int_\Omega h(x, y) \, \mathrm{d}x \, \mathrm{d}y = 1, \qquad u(x, y), h(x, y) \geq 0, \qquad h(x, y) = h(-x, -y).$$

The first condition is imposed by the function $\mathcal{C}_1$ which normalises the blur function, dividing it by its integral over the domain, and is given by

$$\mathcal{C}_1(h(x, y)) = \frac{h(x, y)}{\int_\Omega h(x, y) \, \mathrm{d}x \, \mathrm{d}y}.$$

The second condition is imposed by projections $\mathbb{P}(h)$ and $\mathbb{P}(u)$ of the functions onto a non-negative range by setting any negative values equal to zero, given by

$$\mathbb{P}(h(x, y)) = \begin{cases} h(x, y) & \text{if } h(x, y) > 0 \\ 0 & \text{otherwise} \end{cases},$$

and similarly for u. The final condition is imposed by the function $\mathcal{C}_2$ which maps h to

a rotationally symmetric approximation given by

$$\mathcal{C}_2(h(x,y)) = \frac{h(x,y) + h(-x,-y)}{2}.$$

These conditions are imposed after each update of the blur and image functions.

Boundary Conditions

We now consider boundary conditions of the image and blur function, as considered in §4.1.4. Recall that the total variation operator $L(\boldsymbol{u})$ becomes asymmetric under Neumann boundary conditions. We therefore replace this operator by the sum of symmetric and asymmetric components $L(\boldsymbol{u}) = L_S^+(\boldsymbol{u}) + L_S^-(\boldsymbol{u})$, where $L_S^{\pm}(\boldsymbol{u})$ is defined above, and modify the solution algorithm to use a fixed-point lagging technique. Incorporating this idea into the Euler-Lagrange equations (4.4)—(4.5) yields the modified equations

$$(H^\top H + \alpha L_S^+(\boldsymbol{u})\boldsymbol{u} = H^\top \boldsymbol{z} - \alpha_1 L_S^-(\boldsymbol{u})\boldsymbol{u}, \tag{4.6}$$

$$(U^\top U + \alpha L_S^+(\boldsymbol{h})\boldsymbol{h} = U^\top \boldsymbol{z} - \alpha_2 L_S^-(\boldsymbol{h})\boldsymbol{h}, \tag{4.7}$$

where $\boldsymbol{u} = \mathbb{M}(u)$, $\boldsymbol{h} = \mathbb{M}(h)$ and $\boldsymbol{z} = \mathbb{M}(z)$ are vectorised counterparts of the $m \times n$ discretised image u, blur function h and received image z respectfully given by the mapping $\mathbb{M}(u_{i,j}) = \boldsymbol{u}_{i+m(j-1)}$. U and H are dense $mn \times mn$ matrices representing the image and blur function respectively such that $\mathbb{M}^{-1}(U\boldsymbol{h})$ gives the discretisation of the convolution $[u * h](x,y)$ and $\mathbb{M}^{-1}(H\boldsymbol{u})$ gives the discretisation of the convolution $[h * u](x,y)$. In practice, the large and dense matrices U and H are not formed. As above the operation of convolution is implemented by Fourier transforms but for ease of notation we continue to write the convolution as the matrix-vector product.

Algorithm

The resulting algorithm is presented in Algorithm 9. For brevity, the notation i_b^a is used to denote the step $i - 1 + a/b$. Note that there is no requirement for a separate algorithm for the Neumann and Dirichlet cases since the asymmetric component of the tv operator $L(\cdot)$ would simply be equal to zero and the symmetric component would be equal to $L(\cdot)$. The equations (4.6) and (4.7) then reduce to the Dirichlet case and the maximum number of iterations for j in Algorithm 9 should be set to 1.

Experimental results

We include some examples of restoring images from motion blur corruption by Algorithm 9 in Figures 4.5—4.6. In the case of the satellite, we use Dirichlet boundary conditions and in the case of the colour Fundus retina example, we use Neumann boundary conditions. In both cases, the result is reasonable for a blind deblurring model but still still leaves room for improvement.

Algorithm 9 Blind TV Deblurring (BTVD)

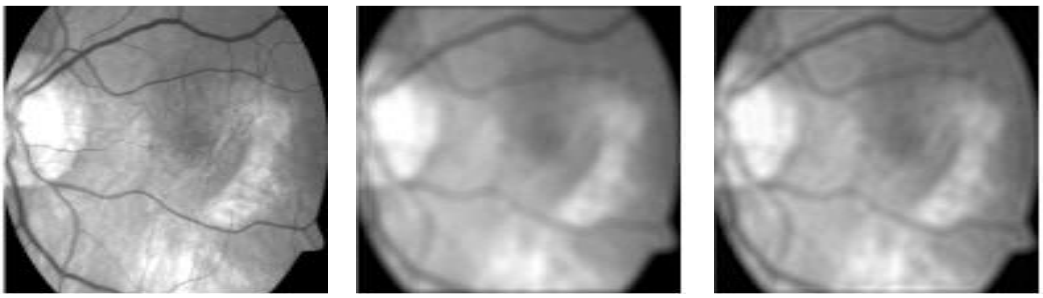
function $(\boldsymbol{h}, \boldsymbol{u}) \leftarrow$BTVD$(\boldsymbol{z}, \alpha_1, \alpha_2, \beta_1, \beta_2, maxit)$
  $\boldsymbol{u}^{(0)} \leftarrow \boldsymbol{z}$
  for $i = 1..maxit$ **do**
    Obtain approximation of the blur function by solving (4.4):
    for $j = 1..maxit$ **do**
      for $k = 1..maxit$ **do**
        Using preconditioned conjugate gradient, solve for $\boldsymbol{h}^{(i_4^1,j,k)}$:

$$\left(U^\top U + \alpha L_S^+\left(\boldsymbol{h}^{(i_4^1,j,k-1)}\right)\right)\boldsymbol{h}^{(i_4^1,j,k)} = U^\top \boldsymbol{z} - \alpha L_S^-\left(\boldsymbol{h}^{(i_4^1,j-1)}\right)\boldsymbol{h}^{(i_4^1,j-1)},$$

      end for
    end for
    Impose $\boldsymbol{h}^{(i_4^2)} = \mathcal{C}_1\left(\boldsymbol{h}^{(i_4^1)}\right)$.
    Impose $\boldsymbol{h}^{(i_4^3)} = \mathbb{P}\left(\boldsymbol{h}^{(i_4^2)}\right)$.
    Impose $\boldsymbol{h}^{(i)} = \mathcal{C}_2\left(\boldsymbol{h}^{(i_4^3)}\right)$.
    Obtain approximation of the image by solving (4.5):
    for $j = 1..maxit$ **do**
      for $k = 1..maxit$ **do**
        Using preconditioned conjugate gradient, solve for $\boldsymbol{u}^{(i_2^1,j,k)}$:

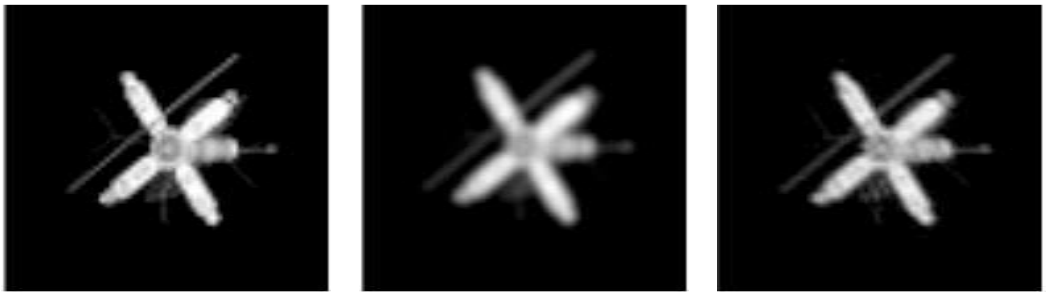
$$\left(H^\top H + \alpha L_S^+\left(\boldsymbol{u}^{(i_2^1,j,k-1)}\right)\right)\boldsymbol{u}^{(i_2^1,j,k)} = H^\top \boldsymbol{z} - \alpha L_S^-\left(\boldsymbol{u}^{(i_2^1,j-1)}\right)\boldsymbol{u}^{(i_2^1,j-1)}$$

      end for
    end for
    Impose $\boldsymbol{u}^{(i)} = \mathbb{P}\left(\boldsymbol{u}^{(i_2^1)}\right)$.
  end for
  $\boldsymbol{h} \leftarrow \boldsymbol{h}^{(i)}$
  $\boldsymbol{u} \leftarrow \boldsymbol{u}^{(i)}$
end function



(a) True Image (b) Received Image (c) Restored Image

Figure 4.5: Illustration of the performance of Algorithm 9 with a colour Fundus retina image, using Neumann boundary conditions. From left to right, we have (a) true image, (b) the received image and (c) the restored image. Some improvement can be seen in the restored image which has an improved SNR value (20.79) over the received image (19.15) and an improved PSNR value (27.00) of the received image (25.03).



(a) True Image (b) Received Image (c) Restored Image

Figure 4.6: Illustration of the performance of Algorithm 9 with the satellite image, using Dirichlet boundary conditions. From left to right, we have (a) true image, (b) the received image and (c) the restored image. Some improvement can be seen in the restored image which has an improved SNR value (16.10) over the received image (8.95) and an improved PSNR value (30.23) of the received image (22.74).

4.1.6 Conclusion

In this section, we presented the implementation of 2 models from image deblurring which can be solved quickly using Fourier transforms to calculate the convolution. We have also presented enhancements of these models for retinal images by consideration of boundary conditions and a solution method.

The models presented show promising results for motion blur in the blind case and several types of blur in the non-blind case. In particular, we see that the results of the retina scan deblurring are improved when we use Neumann boundary conditions.

4.2 An Accelerated Deblurring Model by Variable Splitting

In the previous section, we have presented some approaches to tackle the problem of restoring an image from blur degradation in the cases of known and unknown blur functions. We have seen that these models can achieve good approximations of the true image, which may be improved by considering the boundary conditions. While such models can be implemented with reasonable speed, particularly in the non-blind case of known blur function where only the image is to be approximated, the solution speed is relative to the image size. Even for example images of size 256×256, as used in the previous section, deconvolution models can be slow to solve and the cpu time increases considerably for larger images and more complex problems such as blind deblurring.

In this section, we present a method of acceleration for an image deconvolution algorithm implemented by a separation of variables in the functional. This allows us to avoid solving a dense and non-linear $mn \times mn$ system resulting from the minimisation of the functional, by replacing the system with an equivalent collection of two others, one which is dense but linear and a second which is non-linear but sparse. The solution

of these sub-problems, which should be equivalent to that of the original problem, may then be obtained with greater efficiency.

4.2.1 Dense and Non-Linear System

We consider the problem of restoring an image from blur corruption by total variation deblurring, introduced in §3.3.5, which results in the minimisation problem

$$\min_u \{f(u(\boldsymbol{x}))\} \equiv \min_u \{F(u(\boldsymbol{x})) + \alpha J(u(\boldsymbol{x}))\} \tag{4.8}$$

where α is a regularisation parameter, $F(u(\boldsymbol{x}))$ is a data fitting term which aims to find the image whose convolution with the (known) blur function is close to the received data in the sense that the L^2-norm of the difference is small, and $J(u(\boldsymbol{x}))$ is a regularisation function which imposes a smoothness constrained on the restored image. In this case, the data fitting and smoothness terms are given by

$$F(u(\boldsymbol{x})) = \int_\Omega ([k * u](\boldsymbol{x}) - z(\boldsymbol{x}))^2 \,\mathrm{d}\boldsymbol{x}, \qquad J(u(\boldsymbol{x})) = \int_\Omega |\nabla u(\boldsymbol{x})|_\beta, \tag{4.9}$$

where $k(\boldsymbol{x})$ is the blur degradation function, $z(\boldsymbol{x})$ is the received image. $J(u(\boldsymbol{x}))$ is a smooth approximation to the total variation semi-norm of the image with

$$|\nabla u(\boldsymbol{x})|_\beta = \sqrt{|\nabla u(\boldsymbol{x})|^2 + \beta}, \quad \beta \in \mathbb{R},$$

where β is a small positive parameter. The minimisation of the functional f given in equations (4.8)—(4.9) with respect to the image results in the Euler-Lagrange equation

$$G(u(\boldsymbol{x})) + \alpha L(u(\boldsymbol{x})) = 0 \tag{4.10}$$

where the functions $G(u(\boldsymbol{x}))$ and $L(u(\boldsymbol{x}))$ result from the minimisation of the functions $F(u(\boldsymbol{x}))$ and $J(u(\boldsymbol{x}))$ respectfully defined above in (4.9) and are given by

$$G(u(\boldsymbol{x})) = k^\dagger(\boldsymbol{x}) * (h(\boldsymbol{x}) * u(\boldsymbol{x}) - z(\boldsymbol{x})), \qquad L(u(\boldsymbol{x})) = -\nabla \cdot \left(\frac{\nabla u(\boldsymbol{x})}{|\nabla u(\boldsymbol{x})|_\beta} \right),$$

where $k^\dagger(\boldsymbol{x})$ is the adjoint of $k(\boldsymbol{x})$. In order to find a solution to (4.10), we must solve a large $mn \times mn$ system which is dense and non-linear. It can be noted that the equation for $G(u(\boldsymbol{x}))$ contains two convolutions but is linear in terms of $u(\boldsymbol{x})$, while the equation for the regularisation term $L(u(\boldsymbol{x}))$ is non-linear and results in a $mn \times mn$ system which is large but sparse. It would be useful therefore to consider these equations separately.

4.2.2 Separating Deblurring and Denoising

We aim to separate the large, dense and non-linear problem (4.10) by forming a new joint minimisation problem which is minimised by the solution of one system which is dense but linear and another which is non-linear but sparse. We achieve this by modifying the equation (4.8), introducing a function $v(\boldsymbol{x})$ which replaces the image

$u(\boldsymbol{x})$ in the fitting term. We add a constraint that the two functions should be equal $u(\boldsymbol{x}) = v(\boldsymbol{x})$ so that our new problem given by

$$\min_{u,v} \left\{ f(u,v) = \frac{1}{2} \int_\Omega \left([k * v](\boldsymbol{x}) - z(\boldsymbol{x})\right)^2 \mathrm{d}\boldsymbol{x} + \alpha \int_\Omega |\nabla u(\boldsymbol{x})|_\beta \right\} \tag{4.11}$$

$$\text{s.t. } u(\boldsymbol{x}) - v(\boldsymbol{x}) = 0. \tag{4.12}$$

is equivalent to (4.8)—(4.9).

There remains the question of how to incorporate the condition (4.12) into a solution strategy. Simply equating one with the other at each iteration by imposing

$$u(\boldsymbol{x}) := v(\boldsymbol{x}) \qquad \text{or} \qquad v(\boldsymbol{x}) := u(\boldsymbol{x})$$

would result in either minimising the data fitting term regardless of the regularisation or vice versa, so is not an acceptable method of implementation. Alternatively, we may impose

$$u(\boldsymbol{x}) := \frac{u(\boldsymbol{x}) + v(\boldsymbol{x})}{2}, \qquad v(\boldsymbol{x}) := \frac{u(\boldsymbol{x}) + v(\boldsymbol{x})}{2},$$

which would satisfy the condition but would mean imposing a potentially significant modification to the functions $u(\boldsymbol{x})$ and $v(\boldsymbol{x})$ at each iteration since they may differ considerably. We therefore aim to minimise the distance between the functions $u(\boldsymbol{x})$ and $v(\boldsymbol{x})$ given by the L^2-norm of their difference

$$\|u(\boldsymbol{x}) - v(\boldsymbol{x})\|^2_{L^2(\Omega)} = \int_\Omega \left(u(\boldsymbol{x}) - v(\boldsymbol{x})\right)^2 \mathrm{d}\boldsymbol{x}. \tag{4.13}$$

Incorporating this in the functional, we introduce a parameter $\gamma \in \mathbb{R}_{>0}$ and present a relaxation of the the problem (4.8)—(4.9) given by the minimisation of

$$f(u,v) := \frac{1}{2} \int_\Omega \left([k * v](\boldsymbol{x}) - z(\boldsymbol{x})\right) \mathrm{d}\boldsymbol{x} + \alpha \int_\Omega |\nabla u(\boldsymbol{x})| + \frac{\gamma}{2} \|u(\boldsymbol{x}) - v(\boldsymbol{x})\|^2_{L^2(\Omega)}. \tag{4.14}$$

where the condition (4.12) is imposed implicitly by the minimisation of the distance between the functions $u(\boldsymbol{x})$ and $v(\boldsymbol{x})$.

We solve this problem by alternate minimisation of the arguments. That is, we make an initial estimate of the image $u^{(0)}(\boldsymbol{x})$ given by the received image $z(\boldsymbol{x})$. Then, keeping the function $u(\boldsymbol{x})$ fixed, we minimise (4.14) with respect to the function $v(\boldsymbol{x})$, and similarly for the subsequent minimisation with respect to $u(\boldsymbol{x})$. We continue this until the residual of (4.14) is sufficiently small. At a general iteration step ℓ, we carry out the following

$$v^{(\ell)}(\boldsymbol{x}) = \min_v \left\{ f\left(u^{(\ell-1)}(\boldsymbol{x}), v^{(\ell-1)}(\boldsymbol{x}) \,|\, u^{(\ell-1)}(\boldsymbol{x})\right)\right\},$$
$$u^{(\ell)}(\boldsymbol{x}) = \min_u \left\{ f\left(u^{(\ell-1)}(\boldsymbol{x}), v^{(\ell)}(\boldsymbol{x}) \,|\, v^{(\ell)}(\boldsymbol{x})\right)\right\}.$$

In order to calculate the minimisation of (4.14), we derive the Euler-Lagrange equations

as follows, first minimising the functional with respect to u:

$$\begin{aligned}
\frac{\partial}{\partial\varepsilon}f(u+\varepsilon\phi,v)\bigg|_{\varepsilon=0} &= \frac{\partial}{\partial\varepsilon}\int_\Omega\left(\frac{1}{2}\left(k*v-z\right)+\alpha|\nabla(u+\varepsilon\phi)|+\frac{\gamma}{2}\left(u+\varepsilon\phi-v\right)^2\right)\mathrm{d}\boldsymbol{x}\bigg|_{\varepsilon=0}\\
&= \int_\Omega\frac{\partial}{\partial\varepsilon}\left(\frac{1}{2}\left(k*v-z\right)+\alpha|\nabla(u+\varepsilon\phi)|+\frac{\gamma}{2}\left(u+\varepsilon\phi-v\right)^2\right)\mathrm{d}\boldsymbol{x}\bigg|_{\varepsilon=0}\\
&= \int_\Omega\frac{\partial}{\partial\varepsilon}\left(\alpha|\nabla(u+\varepsilon\phi)|+\frac{\gamma}{2}\left(u+\varepsilon\phi-v\right)^2\right)\mathrm{d}\boldsymbol{x}\bigg|_{\varepsilon=0}\\
&= \int_\Omega\frac{\partial}{\partial\varepsilon}\left(\alpha|\nabla(u+\varepsilon\phi)|\right)\mathrm{d}\boldsymbol{x}\bigg|_{\varepsilon=0}+\int_\Omega\frac{\partial}{\partial\varepsilon}\left(\frac{\gamma}{2}\left(u+\varepsilon\phi-v\right)^2\right)\mathrm{d}\boldsymbol{x}\bigg|_{\varepsilon=0}\\
&= \alpha\int_\Gamma\phi\frac{\nabla u}{|\nabla u|}\cdot\tilde{n}\,\mathrm{d}\boldsymbol{x}+\int_\Omega\gamma\left(u+\varepsilon\phi-v\right)\phi-\alpha\nabla\cdot\frac{\nabla u}{|\nabla u|}\phi\,\mathrm{d}\boldsymbol{x}\bigg|_{\varepsilon=0}\\
&= \alpha\int_\Gamma\phi\frac{\nabla u}{|\nabla u|}\cdot\tilde{n}\,\mathrm{d}\boldsymbol{x}+\int_\Omega\left(\gamma\left(u-v\right)-\alpha\nabla\cdot\frac{\nabla u}{|\nabla u|}\right)\phi\,\mathrm{d}\boldsymbol{x}. \qquad (4.15)
\end{aligned}$$

We now present the minimisation of the functional (4.14) with respect to the function v:

$$\begin{aligned}
\frac{\partial}{\partial\varepsilon}f(u,v+\varepsilon\phi)\bigg|_{\varepsilon=0} &= \frac{\partial}{\partial\varepsilon}\int_\Omega\left(\frac{1}{2}\left(k*(v+\varepsilon\phi)-z\right)+\alpha|\nabla u|+\frac{\gamma}{2}\left(u-(v+\varepsilon\phi)\right)^2\right)\mathrm{d}\boldsymbol{x}\bigg|_{\varepsilon=0}\\
&= \int_\Omega\frac{\partial}{\partial\varepsilon}\left(\frac{1}{2}\left(k*(v+\varepsilon\phi)-z\right)+\alpha|\nabla u|+\frac{\gamma}{2}\left(u-(v+\varepsilon\phi)\right)^2\right)\mathrm{d}\boldsymbol{x}\bigg|_{\varepsilon=0}\\
&= \int_\Omega\frac{\partial}{\partial\varepsilon}\left(\frac{1}{2}\left(k*(v+\varepsilon\phi)-z\right)+\frac{\gamma}{2}\left(u-(v+\varepsilon\phi)\right)^2\right)\mathrm{d}\boldsymbol{x}\bigg|_{\varepsilon=0}\\
&= \int_\Omega\left(k^\dagger*\left(k*v-z\right)\phi+\frac{\gamma}{2}\frac{\partial}{\partial\varepsilon}\left(u-(v+\varepsilon\phi)\right)^2\right)\mathrm{d}\boldsymbol{x}\bigg|_{\varepsilon=0}\\
&= \int_\Omega\left(k^\dagger*\left(k*v-z\right)\phi+\gamma\left(v+\varepsilon\phi-u\right)\phi\right)\mathrm{d}\boldsymbol{x}\bigg|_{\varepsilon=0}\\
&= \int_\Omega\left(k^\dagger*\left(k*v-z\right)+\gamma\left(v-u\right)\right)\phi\,\mathrm{d}\boldsymbol{x}. \qquad (4.16)
\end{aligned}$$

We may now present, from (4.15) and (4.16) the Euler-Lagrange equations (4.17) and (4.18) which are to be solved for u and v in order to minimise the functional (4.14). Note that we have made use of the associative property of convolution and the δ-function which is equal to one for $\boldsymbol{x}=\boldsymbol{0}$ and zero otherwise.

$$\gamma\left(u(\boldsymbol{x})-v(\boldsymbol{x})\right)-\alpha\nabla\cdot\frac{\nabla u(\boldsymbol{x})}{|\nabla u(\boldsymbol{x})|_\beta}=0, \qquad (4.17)$$

$$\Big(k^\dagger(\boldsymbol{x})*k(\boldsymbol{x})+\gamma\delta(\boldsymbol{x})\Big)*v(\boldsymbol{x})=k^\dagger(\boldsymbol{x})*z(\boldsymbol{x})+\gamma u(\boldsymbol{x}). \qquad (4.18)$$

In order to find a solution, we first discretise the Euler-Lagrange equations over a mesh of size $m\times n$ and consider appropriate solution methods. Equation (4.18) may be solved efficiently using discrete Fourier transforms or conjugate gradient. In order

to solve (4.17), we use time marching given by

$$u_t = -\nabla_u f(u,v), \qquad \nabla_u f(u,v) = \gamma\left(u(\boldsymbol{x}) - v(\boldsymbol{x})\right) - \alpha\nabla \cdot \frac{\nabla u(\boldsymbol{x})}{|\nabla u(\boldsymbol{x})|_\beta}.$$

Discretising with respect to time, we obtain an iterative procedure. We select a step-size Δ_t and an initial estimate of $v(\boldsymbol{x})$ given by $u(\boldsymbol{x})$ and proceed to carry out the following iterations given, at time t, by

$$u^{(t)} = u^{(t-1)}(\boldsymbol{x}) + \Delta_t\left(\alpha\nabla \cdot \frac{\nabla u^{(t-1)}(\boldsymbol{x})}{|\nabla u^{(t-1)}(\boldsymbol{x})|_\beta} - \gamma\left(u^{(t-1)}(\boldsymbol{x}) - v(\boldsymbol{x})\right)\right).$$

We continue until the residual is sufficiently small. We present the complete algorithm for this method below in Algorithm 10.

Algorithm 10 Split Variable Deblurring (SVaD)

function $u \leftarrow$SVaD$(z, k, \alpha, \gamma, tol, maxit)$
  $v^{(0)} \leftarrow z$
  for $\ell = 1, \ldots, maxit$ **do**
    Solve (4.17) for $u^{(\ell)}$:
    for $t = 1, \ldots, maxit$ **do**
      Evaluate:

$$E^{(\ell,t)}(\boldsymbol{x}) = \gamma\left(u^{(\ell,t-1)}(\boldsymbol{x}) - v^{(\ell)}(\boldsymbol{x})\right) - \alpha\nabla \cdot \frac{\nabla u^{(\ell,t-1)}(\boldsymbol{x})}{|\nabla u^{(\ell,t-1)}(\boldsymbol{x})|}$$

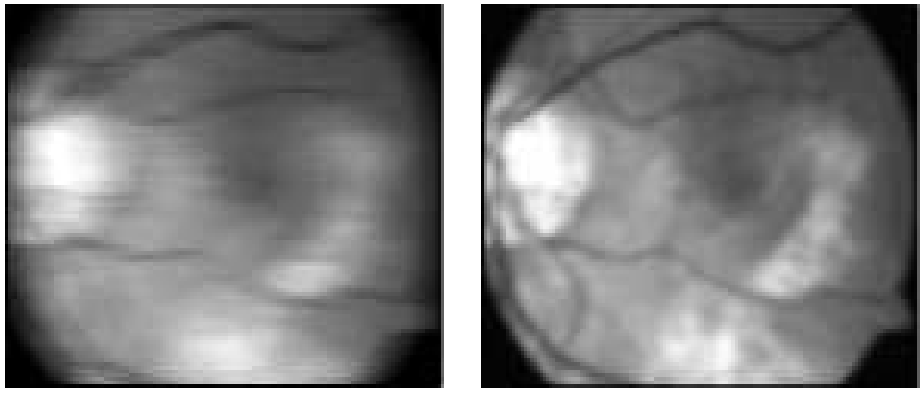
      $u^{(\ell,t)} = u^{(\ell,t-1)}(\boldsymbol{x}) - \Delta_t E^{(\ell,t)}(\boldsymbol{x})$
      if $\left\|E^{(\ell,t)}(\boldsymbol{x}) - E^{(\ell,t-1)}(\boldsymbol{x})\right\|^2_{L^2(\Omega)} \leq tol$ **then**
        break
      end if
    end for
    Solve (4.18) for $v^{(\ell)}$:

$$\left(k^\dagger(\boldsymbol{x}) * k(\boldsymbol{x}) + \gamma\delta(\boldsymbol{x})\right) * v^{(\ell)}(\boldsymbol{x}) = k^\dagger(\boldsymbol{x}) * z(\boldsymbol{x}) + \gamma u^{(\ell)}(\boldsymbol{x})$$

    if $\|u - v\|^2_{L^2(\Omega)} \leq tol$ **then**
      break
    end if
  end for
  $u(\boldsymbol{x}) \leftarrow u^\ell(\boldsymbol{x})$
end function

4.2.3 Experimental Results

We include examples of restoring satellite and colour Fundus retina images from motion and Gaussian blur degradation by Algorithm 10 in Figures 4.7—4.9. In all cases, the image is restored to good quality with low cpu time.



(a) Received Image (b) Restored Image

Figure 4.7: Illustration of the performance of Algorithm 10 with a colour Fundus Retina image corrupted by motion blur. On the left is (a) the received image and on the right (b) the restored image, obtained with a cpu time of 1.48. It can be noticed that the restored image with psnr 27.145 is an improvement on the received image (a) which has psnr 22.847.

4.2.4 Blind Deblurring

In this section, we attempt to implement this idea of variable splitting in the case deblurring without knowledge of the blur function. Following §3.4.2, we attempt to simultaneously restore both the blur function and the true image by the alternate minimisation of a cost functional given some additional constraints. We achieve this by forming a functional consisting of a fitting term, which aims to minimise the distance between the received image and the convolution of the restored image and blur function. We also add smoothness constraints on the image and blur function by imposing regularisation. We then attempt to find the pair of functions $(u(\boldsymbol{x}), h(\boldsymbol{x}))$ which minimise the functional subject to the additional constraints which ensure that both the blur function and image take non-negative values, and that the blur function is rotationally symmetric and has a unit integral.

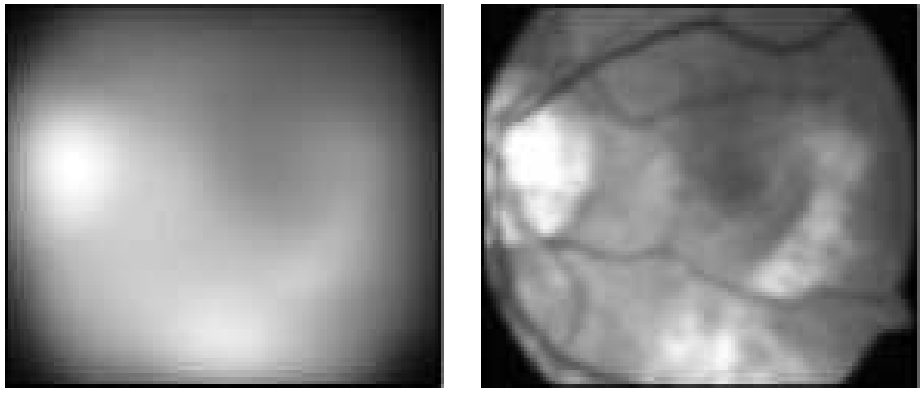
Given these considerations, we modify the functional (4.14) with the addition of regularisation of the blur function, resulting in the following minimisation problem

$$\min_{u,v,h}\Bigg\{f(u,v,h) = \frac{1}{2}\int_\Omega \left([k*v](\boldsymbol{x}) - z(\boldsymbol{x})\right)\mathrm{d}\boldsymbol{x} + \alpha_1\int_\Omega |\nabla u(\boldsymbol{x})| + \frac{\gamma}{2}\left\|u(\boldsymbol{x}) - v(\boldsymbol{x})\right\|^2_{L^2(\Omega)} + \alpha_2\int_\Omega |\nabla h(\boldsymbol{x})|\Bigg\}. \tag{4.19}$$

subject to the constraints

$$h(\boldsymbol{x}) \succeq 0, \qquad u(\boldsymbol{x}) \succeq 0, \qquad \int_\Omega h(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} = 1, \qquad h(\boldsymbol{x}) = h(-\boldsymbol{x}). \tag{4.20}$$

Note that we do not explicitly impose the non-negativity constraint on the function $v(\boldsymbol{x})$ since the implicit similarity constraint (4.13) should lead to the non-negativity



(a) Received Image (b) Restored Image

Figure 4.8: Illustration of the performance of Algorithm 10 with a colour Fundus Retina image corrupted by Gaussian blur. On the left is (a) the received image and on the right (b) the restored image, obtained with a cpu time of 1.63. It can be noticed that the restored image with psnr 27.003 is an improvement on the received image (a) which has psnr 17.862.

constraint being automatically satisfied.

We can note that the minimisation of (4.20) with respect to the blur function $h(\boldsymbol{x})$ results in a dense and non-linear sub-problem. We would to resolve this into two systems, one which is dense and linear, and another which is non-linear but sparse in a similar manner to §4.2.2. To this end, we introduce a new function $k(\boldsymbol{x})$ which replaces the function $h(\boldsymbol{x})$ in the fitting term and give an additional constraint which requires the functions $k(\boldsymbol{x})$ and $h(\boldsymbol{x})$ to be similar. We imposing this constraint by minimising the L^2-norm of the difference

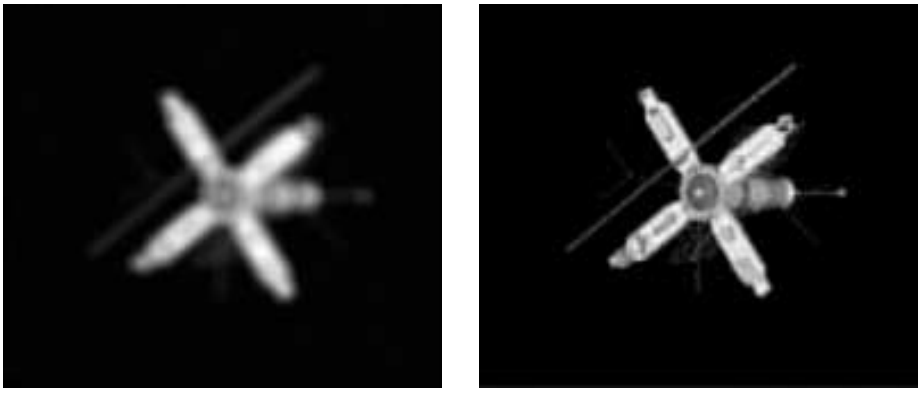
$$\|h(\boldsymbol{x}) - k(\boldsymbol{x})\|^2_{L^2(\Omega)} = \int_\Omega (h(\boldsymbol{x}) - k(\boldsymbol{x}))^2 \, \mathrm{d}\boldsymbol{x}, \tag{4.21}$$

with respect to the blur function $h(\boldsymbol{x})$ and the function $k(\boldsymbol{x})$. We then modify the cost functional (4.19) to include this constraint resulting in the new problem

$$\begin{aligned} \min_{u,v,h,k} \Bigg\{ f(u,v,h,k) = \frac{1}{2} \int_\Omega ([k * v](\boldsymbol{x}) - z(\boldsymbol{x}))^2 \, \mathrm{d}\boldsymbol{x} + \frac{\gamma_1}{2} \|u(\boldsymbol{x}) - v(\boldsymbol{x})\|^2_{L^2(\Omega)} \\ + \alpha_1 \int_\Omega |\nabla u(\boldsymbol{x})|_\beta + \frac{\gamma_2}{2} \|h(\boldsymbol{x}) - k(\boldsymbol{x})\|^2_{L^2(\Omega)} + \alpha_2 \int_\Omega |\nabla h(\boldsymbol{x})|_\beta \Bigg\}, \end{aligned} \tag{4.22}$$

where $\alpha_1, \alpha_2 \in \mathbb{R} > 0$ are small positive parameters which may be chosen to control the amount of smoothing of the image kernel, and $\gamma_1, \gamma_2 \in \mathbb{R}_{>0}$ are small positive parameters which measure the trade-off between the fitting of the two image and point spread functions and the data fitting and smoothness terms.

We aim to find functions $u(\boldsymbol{x})$, $v(\boldsymbol{x})$, $h(\boldsymbol{x})$ and $k(\boldsymbol{x})$ which minimise (4.22). We proceed with the alternate minimisation of the arguments. Minimising (4.22) with respect to $u(\boldsymbol{x})$, $v(\boldsymbol{x})$, $h(\boldsymbol{x})$ and $k(\boldsymbol{x})$ respectively, we obtain the following Euler Lagrange



(a) Received Image (b) Restored Image

Figure 4.9: Illustration of the performance of Algorithm 10 with the satellite image corrupted by motion blur. On the left is (a) the received image and on the right (b) the restored image. It can be noticed that the restored image with psnr 147.510 is an improvement on the received image (a) which has psnr 142.688.

equations

$$\gamma_1(u(\boldsymbol{x}) - v(\boldsymbol{x})) - \alpha_1 \nabla \cdot \left(\frac{\nabla u(\boldsymbol{x})}{|\nabla u(\boldsymbol{x})|_\beta} \right) = 0 \tag{4.23}$$

$$\gamma_1(v(\boldsymbol{x}) - u(\boldsymbol{x})) + k(-\boldsymbol{x}) * (k(\boldsymbol{x}) * v(\boldsymbol{x}) - z(\boldsymbol{x})) = 0 \tag{4.24}$$

$$\gamma_2(h(\boldsymbol{x}) - k(\boldsymbol{x})) - \alpha_2 \nabla \cdot \left(\frac{\nabla h(\boldsymbol{x})}{|\nabla h(\boldsymbol{x})|_\beta} \right) = 0 \tag{4.25}$$

$$\gamma_2(k(\boldsymbol{x}) - h(\boldsymbol{x})) + v(-\boldsymbol{x}) * (v(\boldsymbol{x}) * k(\boldsymbol{x}) - z(\boldsymbol{x})) = 0 \tag{4.26}$$

4.2.5 Solution Algorithm

We proceed to minimise (4.22) by minimising alternately with respect to the arguments and implementing the constraints (4.20) at each step. An overall algorithm is given in Algorithm 11. We use the notation ℓ_b^a to denote the step $\ell - 1 + a/b$ and we use the functions

$$\mathbb{P}(h(\boldsymbol{x})) = \begin{cases} h(\boldsymbol{x}) & \text{if } h(\boldsymbol{x}) > 0 \\ 0 & \text{otherwise,} \end{cases}$$

$$\mathcal{C}_1(h(\boldsymbol{x})) = \frac{h(\boldsymbol{x}) + h(-\boldsymbol{x})}{2}, \qquad \mathcal{C}_2(h(\boldsymbol{x})) = \frac{h(\boldsymbol{x})}{\int_\Omega h(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}},$$

in order to enforce the explicit constraints.

4.2.6 Experimental Results

We present experimental results of blind restoration of the satellite and retina images corrupted by motion blur using Algorithm 11 in Figures (4.10) and (4.11). In both cases the images are improved in a small amount of cpu time, achieving reasonable results for the satellite image. However the heavily blurred retina image retains much of the degradation and the blur function is not well approximated.

Algorithm 11 Blind Split Variable Deblurring (BSVaD)

function $(h, u) \leftarrow$BSVaD$(z, h_{initial}(\boldsymbol{x}), \alpha_1, \alpha_2, \gamma_1, \gamma_2, tol, maxit)$
  $u^{(0)}(\boldsymbol{x}) \leftarrow z(\boldsymbol{x})$
  $h^{(0)}(\boldsymbol{x}) \leftarrow h_{initial}(\boldsymbol{x})$
  for $\ell \leftarrow 1$ **to** $maxit$ **do**
    repeat
      Solve for $h^{(\ell_4^1)}(\boldsymbol{x})$:

$$\gamma_2 \left(h^{(\ell_4^1)}(\boldsymbol{x}) - k^{(\ell-1)}(\boldsymbol{x}) \right) - \alpha \nabla \cdot \left(\frac{\nabla h^{(\ell_4^1)}(\boldsymbol{x})}{|\nabla h^{(\ell_4^1)}(\boldsymbol{x})|} \right) = 0$$

      Solve for $k^{(\ell)}(\boldsymbol{x})$:

$$v^{(\ell-1)}(-\boldsymbol{x}) * \left(v^{(\ell-1)}(\boldsymbol{x}) * k^{(\ell)}(\boldsymbol{x}) - z(\boldsymbol{x}) \right) + \gamma_2 (k^{(\ell)}(\boldsymbol{x}) - h^{(\ell_4^1)}(\boldsymbol{x})) = 0$$

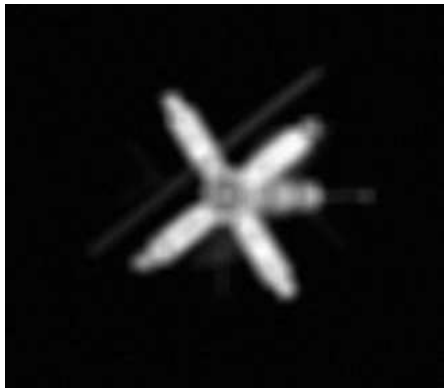
    until $\left\| h^{(\ell_4^1)}(\boldsymbol{x}) - k^{(\ell)}(\boldsymbol{x}) \right\|^2_{L^2(\Omega)} \leq tol$
    Impose: $h^{(\ell_4^2)}(\boldsymbol{x}) \leftarrow \mathbb{P}\left(h^{(\ell_4^1)}(\boldsymbol{x}) \right)$
    Impose: $h^{(\ell_4^3)}(\boldsymbol{x}) \leftarrow \mathcal{C}_1\left(h^{(\ell_4^2)}(\boldsymbol{x}) \right)$
    Impose: $h^{(\ell)}(\boldsymbol{x}) \leftarrow \mathcal{C}_2\left(h^{(\ell_4^3)}(\boldsymbol{x}) \right)$
    repeat
      Solve for $u^{(\ell_2^1)}(\boldsymbol{x})$:

$$\gamma (u^{(\ell_2^1)}(\boldsymbol{x}) - v^{(\ell-1)}(\boldsymbol{x})) - \alpha \nabla \cdot \left(\frac{\nabla u^{(\ell_2^1)}(\boldsymbol{x})}{|\nabla u^{(\ell_2^1)}(\boldsymbol{x})|} \right) = 0$$

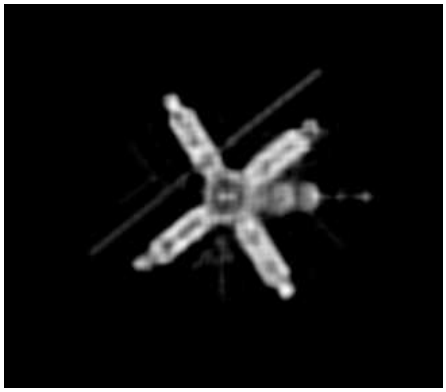
      Solve for $v^{(\ell)}(\boldsymbol{x})$:

$$k^{(\ell)}(-\boldsymbol{x}) * \left(k^{(\ell)}(\boldsymbol{x}) * v^{(\ell)}(\boldsymbol{x}) - z(\boldsymbol{x}) \right) + \gamma (v^{(\ell)}(\boldsymbol{x}) - u^{(\ell_2^1)}(\boldsymbol{x})) = 0$$

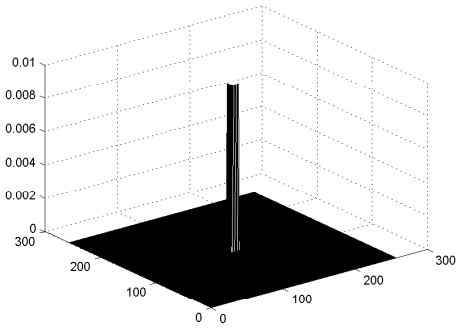
    until $\left\| u^{(\ell_2^1)}(\boldsymbol{x}) - v^{(\ell)}(\boldsymbol{x}) \right\|^2_{L^2(\Omega)} \leq tol$
    Impose: $u^{(\ell)}(\boldsymbol{x}) \leftarrow \mathbb{P}\left(u^{(\ell_2^1)}(\boldsymbol{x}) \right)$
  end for
  $h(\boldsymbol{x}) \leftarrow h^{(\ell)}(\boldsymbol{x})$
  $u(\boldsymbol{x}) \leftarrow u^{(\ell)}(\boldsymbol{x})$
end function



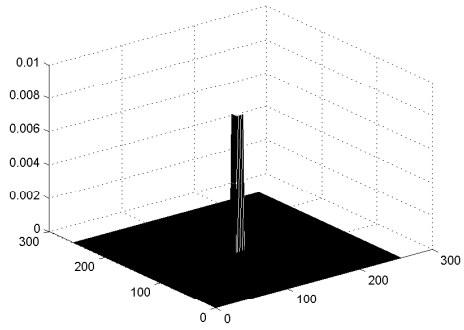
(a) Received Image



(b) Restored Image



(c) Initial Blur Function

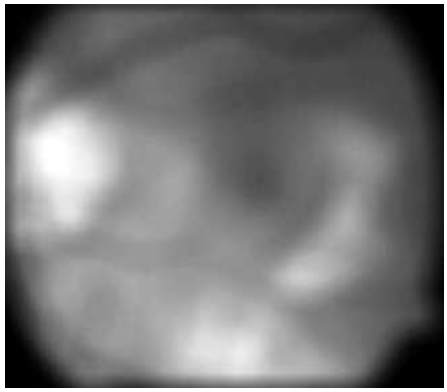


(d) Approximated Blur Function

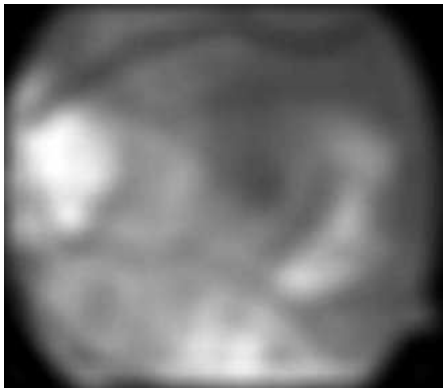
Figure 4.10: Illustration of the performance of Algorithm 11 with the satellite image corrupted by out of focus blur. The cpu time taken to obtain the restored image (b) given the received image (a) and to obtain the approximation of the blur function (d) from the initial estimate (c) is 16.71.

4.2.7 Conclusion

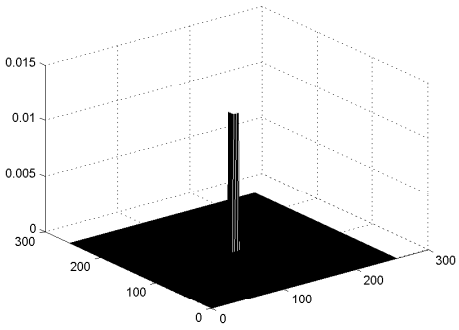
In this section, we have presented an accelerated model for restoring images from blur corruption by separating the non-linearity from the density of the system and iteratively finding solutions of the two resulting systems, each of which can be solved quickly. Experimental results demonstrate that the method can work well, particularly in the case of non-blind deconvolution. In the blind case, where we attempt to simultaneously restore the blur function and the image, the model is efficient and works well in the case of the satellite image but struggles to give an accurate approximation of the blur function in some cases resulting in little improvement in or further degradation of the image. In order to improve this, we may consider alternative methods of blur identification by retaining the dense and non-linear system in the case of the blur function or by estimating the blur function using image statistics (see [17, 30, 75, 76, 92, 123, 131, 168, 191] for examples of work in this area) before proceeding with a deblurring algorithm.



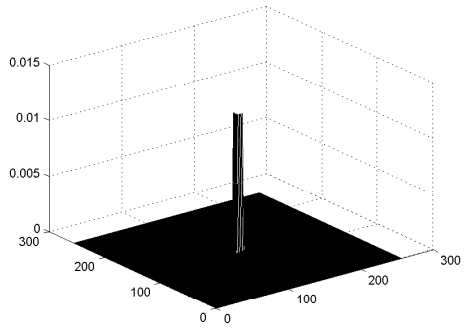
(a) Received Image



(b) Restored Image



(c) Initial Blur Function



(d) Approximated Blur Function

Figure 4.11: Illustration of the performance of Algorithm 11 with the retina image corrupted by out of focus blur. The cpu time taken to obtain the restored image (b) given the received image (a) and to obtain the approximation of the blur function (d) from the initial estimate (c) is 18.77.

4.3 A Constrained Kernel Filtering Model for Blind Deconvolution

In the previous sections, we have presented enhancements to the deblurring models introduced in Chapter 3 by considering boundary conditions and solution speed. In this section, we attempt to improve results obtained from simultaneous reconstruction of a blurred image and the corresponding blur function by introducing modified constraints on the point spread function. We also consider alternative regularisation for the reconstruction of images from degradation by smooth point spread functions such as Gaussian blur.

4.3.1 Simultaneous Blind Image Deblurring

Blind image deblurring tackles the more realistic problem where the blur function is not known. As well as recovering the true image, we aim to simultaneously recover the blur kernel with no a priori information. There exist many algorithms for recovering both the image and the kernel simultaneously. For example, see [78, 121, 120, 216]. In

§3.4.2, we introduced the alternate minimisation blind deblurring model by Chan and Wong [54] given by

$$\min_{u,h}\left\{ f(u,h) = \frac{1}{2}\int_{\Omega} ([h * u](\boldsymbol{x}) - z(\boldsymbol{x}))^2 \,\mathrm{d}\boldsymbol{x} + \alpha_1 \int_{\Omega} |\nabla u(\boldsymbol{x})| + \alpha_2 \int_{\Omega} |\nabla h(\boldsymbol{x})| \right\} \quad (4.27)$$

to recover the image with no a priori information about the blurring function or point spread function.

4.3.2 Optimisation Constraints for Blind Deblurring

Solving the minimisation problem (4.27) does not necessarily yield physical solutions because (4.27) may not have a unique solution. For example, if $(u(\boldsymbol{x}), h(\boldsymbol{x}))$ is a solution, then so are

$$\left(\frac{a_2 h(\boldsymbol{x})}{a_1}, \frac{a_1 u(\boldsymbol{x})}{a_2}\right) \ \forall\, a_1, a_2 \in \mathbb{R}\backslash\{0\}, \qquad (-u(\boldsymbol{x}), -h(\boldsymbol{x}))\,,$$

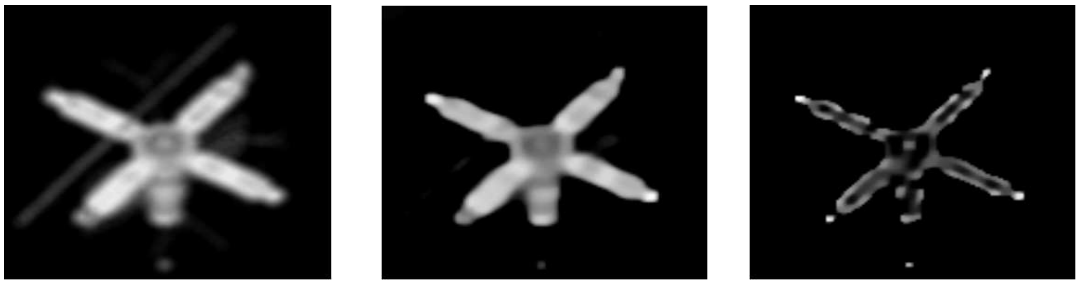
$$(u(x \pm c, y \pm d),\ h(x \mp c, y \mp d)) \ \forall\, c, d \in \mathbb{R}$$

In an attempt to obtain a physical solution, we impose the following conditions on $u(\boldsymbol{x})$ and $h(\boldsymbol{x})$

$$\int_{\Omega} h(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} = 1, \qquad u(\boldsymbol{x}), h(\boldsymbol{x}) \succeq 0, \qquad h(\boldsymbol{x}) = h(-\boldsymbol{x}), \quad (4.28)$$

which are imposed at each iteration. The overall solution algorithm for the minimisation problem (4.27) is given in Algorithm 9 on Page 87.

We include a result of total variation blind deblurring by solving the minimisation problem (4.27) to deblur the satellite image which has been corrupted by motion blur in Figure 4.12. One can notice that while the edges of the image can be recovered, the small details are much more difficult to accurately recover. We will attempt to improve on this result in the next section by considering the optimisation constraints presented earlier.



(a) Received Image (b) Result after 1 Iteration (c) Result after 20 Iterations

Figure 4.12: Illustration of the possible poor performance arising from solving the minimisation problem (4.27) for the blurred satellite image. From left to right, we have (a) the received image of psnr 21.09, (b) the result after one iteration with psnr 15.19, (c) the result after 20 iterations with psnr 20.48. The psnr in both results is lower than that of the received image and the visual quality is clearly diminished.

4.3.3 Alternative Optimisation Constraints

In the previous section, we imposed the optimisation constraints (4.28) at each alternate minimisation stage. In an attempt to improve the quality of the restored image, we consider the following alternatives to such constraints. Firstly, we may remove one or more of the constraints. Secondly, we may impose additional constraints on the image given prior knowledge. Finally, we consider imposing further constraints on the kernel based on observation of the result.

In the application of the given constraints, we impose restrictions on the integral of the kernel, thereby controlling the scale of it but we do not impose any restriction on the scale of the image intensity values, which may allow the intensity values to increase significantly. We therefore attempt to control this by imposing additional constraints on the image by projecting it to the range $[0, \zeta_u]$ by truncation of image intensity values outside of this range where ζ_u is the maximum expected intensity value of the image, but this does not give good results. As demonstrated in Figure 4.13, all of the details of the image are lost and the reconstruction of the overall shape is inaccurate. Removing constraints can offer an improved result but there is more risk of achieving the wrong result. We therefore consider the 3rd alternative in the next section.

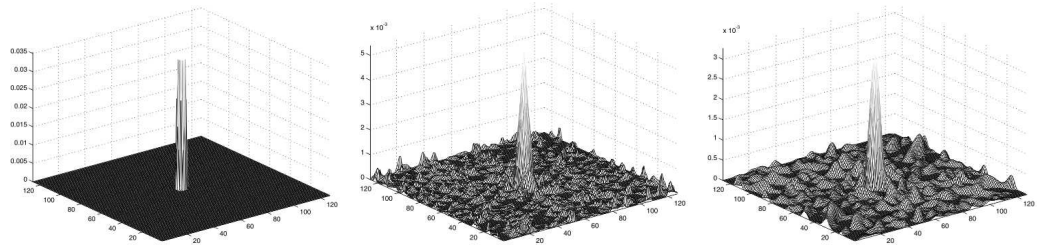
(a) Received Image (b) Approximated Image

Figure 4.13: Illustration of the results obtained after applying harsh constraints to the image by simple projection onto the range $[0, \zeta_u]$ at each iteration On the left is (a) the received image of psnr 21.09 and on the right (b) the approximated image of psnr 13.79.

4.3.4 Intensity Based Constraints for Local Support Kernels

It can be observed, as demonstrated in Figure 4.14 that there exist many small values in the restored kernel arising from the restoration. Such a large number of points which are incorrectly non-zero will have a significant effect on the image. We aim to remove such small values by adjusting our constraints to filter out these values. We replace the constraint $h(\boldsymbol{x}) \succeq 0$ with one which does not permit such small values, given by

$$h(\boldsymbol{x}) \in \mathbb{B} \; \forall \, \boldsymbol{x} \in \Omega, \qquad \mathbb{B} = [0] \cup [\epsilon\zeta_h, \zeta_h]$$



(a) True Kernel (b) Approximated Kernel 1 (c) Approximated Kernel 2

Figure 4.14: Results of the blur function approximation obtained by solving the minimisation problem (4.27) with constraints. From left to right, we have (a) the true kernel we are aiming to approximate, (b) the approximated kernel using the constraints (4.28), and (c) the restored kernel using the constraints (4.28) as well as the image intensity based constraint. It can be noticed that as well as being imprecise there exists much noise in the approximation of the blur function.

where $\epsilon \in (0,1) \subset \mathbb{R}$ is a small positive parameter and ζ_h is the maximal value of the kernel function $h(\boldsymbol{x})$ given by

$$\zeta_h = \max_{\boldsymbol{x}\in\Omega} h(\boldsymbol{x}).$$

This restricts the range of values of the blur function with the aim of reducing the noise in the kernel. In order to implement this restriction, we use the projection

$$\mathbb{P}^L(h(\boldsymbol{x})) = \begin{cases} h(\boldsymbol{x}) & \text{if } h(\boldsymbol{x}) \in \mathbb{B} \\ 0 & \text{otherwise,} \end{cases} \tag{4.29}$$

which sets values in the range $(0, \epsilon\zeta_h)$ equal to zero.

4.3.5 Solution Algorithm

In order to solve this model we minimise (4.27) alternately with respect to the blur function $h(\boldsymbol{x})$ and the image $u(\boldsymbol{x})$. At each iteration, we implement the constraints on the blur function given by (4.29) and

$$\mathcal{C}_1(h(\boldsymbol{x})) = \frac{h(\boldsymbol{x}) + h(-\boldsymbol{x})}{2}, \qquad \mathcal{C}_2(h(\boldsymbol{x})) = \frac{h(\boldsymbol{x})}{\int_\Omega h(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}}, \tag{4.30}$$

and the projection constraint on the image to the non-negative range given by

$$\mathbb{P}(u(\boldsymbol{x})) = \begin{cases} u(\boldsymbol{x}) & \text{if } u(\boldsymbol{x}) \geq 0 \\ 0 & \text{otherwise.} \end{cases} \tag{4.31}$$

We present the solution algorithm for this model in Algorithm 12. We use the notation ℓ_b^a to denote the step $\ell - 1 + a/b$.

Algorithm 12 Deblurring with Local Support Kernels (DLSK)

function $(h(\boldsymbol{x}), u(\boldsymbol{x})) \leftarrow$DLSK$(z(\boldsymbol{x}), h_{initial}(\boldsymbol{x}), \alpha_1, \alpha_2, tol, maxit)$
  $u^{(0)}(\boldsymbol{x}) \leftarrow z(\boldsymbol{x})$
  $h^{(0)}(\boldsymbol{x}) \leftarrow h_{initial}(\boldsymbol{x})$
  for $\ell \leftarrow 1$ **to** $maxit$ **do**
    Solve for $h^{\left(\ell_4^1\right)}(\boldsymbol{x})$:

$$u^{(\ell-1)}(-\boldsymbol{x}) * \left(u^{(\ell-1)}(\boldsymbol{x}) * h^{\left(\ell_4^1\right)}(\boldsymbol{x}) - z(\boldsymbol{x})\right) - \alpha_2 \nabla \cdot \left(\frac{\nabla h^{\left(\ell_4^1\right)}(\boldsymbol{x})}{|\nabla h^{\left(\ell_4^1\right)}(\boldsymbol{x})|_\beta}\right) = 0$$

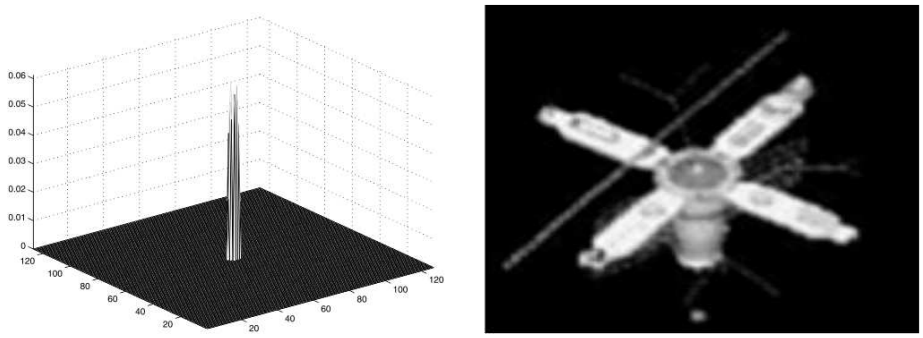
    Impose: $h^{\left(\ell_4^2\right)}(\boldsymbol{x}) \leftarrow \mathbb{P}^L\left(h^{\left(\ell_4^1\right)}(\boldsymbol{x})\right)$
    Impose: $h^{\left(\ell_4^3\right)}(\boldsymbol{x}) \leftarrow \mathcal{C}_1\left(h^{\left(\ell_4^2\right)}(\boldsymbol{x})\right)$
    Impose: $h^{(\ell)}(\boldsymbol{x}) \leftarrow \mathcal{C}_2\left(h^{\left(\ell_4^3\right)}(\boldsymbol{x})\right)$
    Solve for $u^{\left(\ell_2^1\right)}(\boldsymbol{x})$:

$$h^{(\ell)}(-\boldsymbol{x}) * \left(h^{(\ell)}(\boldsymbol{x}) * u^{\left(\ell_2^1\right)}(\boldsymbol{x}) - z(\boldsymbol{x})\right) - \alpha_1 \nabla \cdot \left(\frac{\nabla u^{\left(\ell_2^1\right)}(\boldsymbol{x})}{|\nabla u^{\left(\ell_2^1\right)}(\boldsymbol{x})|_\beta}\right) = 0$$

    Impose: $u^{(\ell)}(\boldsymbol{x}) \leftarrow \mathbb{P}\left(u^{\left(\ell_2^1\right)}(\boldsymbol{x})\right)$
  end for
  $h(\boldsymbol{x}) \leftarrow h^{(\ell)}(\boldsymbol{x})$
  $u(\boldsymbol{x}) \leftarrow u^{(\ell)}(\boldsymbol{x})$
end function

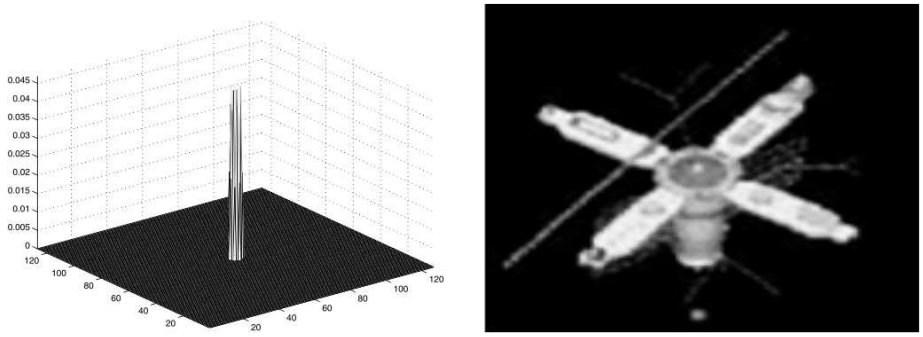
4.3.6 Experimental Results

We include results of restoring by Algorithm 12 satellite and retina images corrupted by out of focus blur in Figures 4.15—4.17. It can be noted that while ϵ must be sufficiently large that it compresses noise, using small values for ϵ improves the accuracy of the restored image. In order to measure our results numerically, we compute the PSNR measure introduced in §4.1.3 for each example.



(a) Approximated Kernel (b) Restored Image

Figure 4.15: Successful restoration of the blurred satellite image shown in Figure 4.12a by Algorithm 12 with $\epsilon = 1/3$. On the left is (a) the approximated blur function and on the right (b) the restored image of psnr 27.28 which is an increase of 6.19dB compared to the received image.

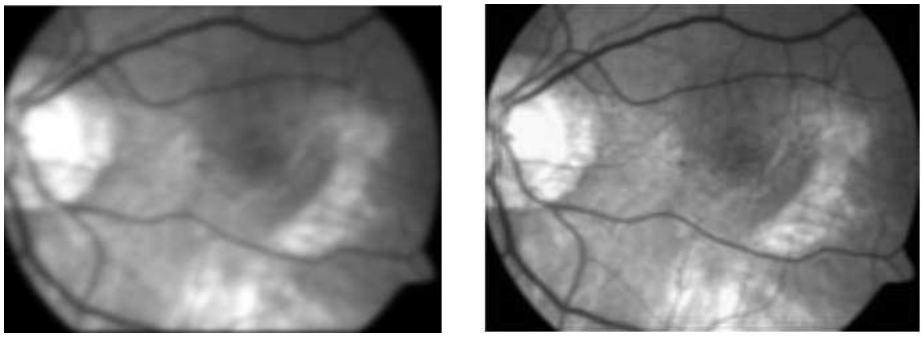


(a) Approximated Kernel (b) Restored Image

Figure 4.16: Successful restoration of the blurred satellite image shown in Figure 4.12a by Algorithm 12 with $\epsilon = 10^{-2}$. On the left is (a) the approximated blur function and on the right (b) the restored image of psnr 30.14 which is an increase of 9.05dB compared to the received image.

4.3.7 A Blind Deblurring Model for Gaussian Blur

In the case of Gaussian blur, total variation regularisation of the kernel can help to approximate the blur function. However, since this function tends to give non-smooth or 'blocky' results, it is typically effective when dealing with kernels which produce out-of-focus or motion blur but it is not necessarily the most appropriate function to use when approximating a smooth kernel. Variation of the smoothing parameter β can



(a) Received Image (b) Restored Image

Figure 4.17: Successful restoration of the blurred retina image by Algorithm 12 with $\epsilon = 10^{-2}$. The restored image (b) is a considerable improvement on the received image (a).

help to smooth out this function, however this can give overly smooth and less accurate results. We therefore consider alternative regularisation in the case of Gaussian blur. We modify our functional as follows, replacing the total variation semi-norm with the H^1 semi-norm of the image given by the L^2-norm of the gradient, given by $\int_\Omega |\nabla h|^2 d\Omega$ which we expect to provide more accurate approximations to smoother kernels.

$$\min_{u,h} \left\{ f(u,h) = \frac{1}{2}||[h*u](\boldsymbol{x}) - z(\boldsymbol{x})||^2_{L^2(\Omega)} + \alpha_1 \int_\Omega |\nabla u(\boldsymbol{x})| + \alpha_2 \int_\Omega |\nabla h(\boldsymbol{x})|^2 \, \mathrm{d}\boldsymbol{x} \right\} \quad (4.32)$$

In order to carry out the minimisation of (4.32), we obtain the following Euler-Lagrange equations which we aim to solve as part of an alternate minimisation scheme.

$$\frac{\partial f}{\partial h} : \; u(-\boldsymbol{x}) * (u(\boldsymbol{x}) * h(\boldsymbol{x}) - z(\boldsymbol{x})) - \alpha_2 \nabla^2 h(\boldsymbol{x}) = 0, \quad (4.33)$$

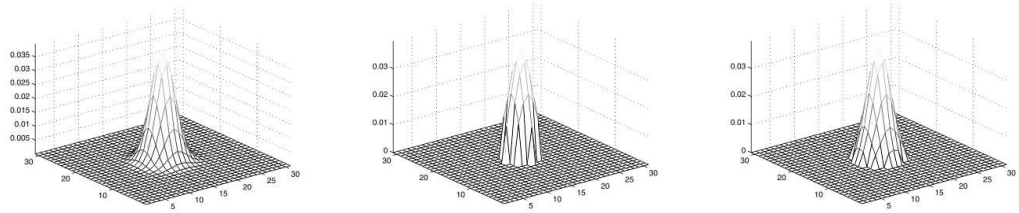
$$\frac{\partial f}{\partial u} : \; h(-\boldsymbol{x}) * (h(\boldsymbol{x}) * u(\boldsymbol{x}) - z(\boldsymbol{x})) - \alpha_1 \nabla \cdot \left(\frac{\nabla u(\boldsymbol{x})}{|\nabla u(\boldsymbol{x})|} \right) = 0. \quad (4.34)$$

4.3.8 Location Based Constraints for Global Support Kernels

The intensity-based constraints which we introduced earlier are not suitable for application to Gaussian kernels. As illustrated in Figure 4.18, Gaussian kernels typically have a smooth base, which is not achievable with the previous intensity-based filter. We therefore use constraints based on the location rather than the values, replacing the optimisation constraints $h(x,y) \succeq 0$ in the original model with $h(x,y) \geq 0$ if $\sqrt{x^2+y^2} \leq \delta$ for $\delta \in [0, \min\{m/2, n/2\}]$ where we are discretising over a mesh of size $m \times n$. This is implemented with the projection

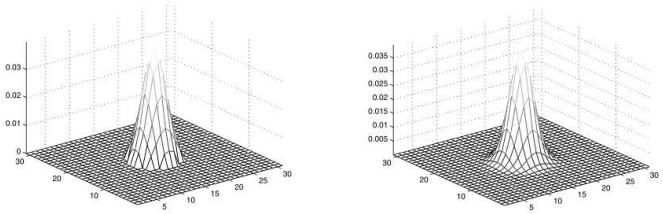
$$\mathbb{P}^G_\delta \left(h(x,y) \right) = \begin{cases} h(x,y) & \text{if } \sqrt{x^2+y^2} \leq \delta \\ 0 & \text{otherwise,} \end{cases} \quad (4.35)$$

which sets values of the blur function outside of the given range equal to zero.



(a) Gaussian Function (b) Constrained, $\epsilon = 1/3$ (c) Constrained, $\epsilon = 10^{-2}$

Figure 4.18: Illustration of the effect of intensity based constraints on a Gaussian function. From left to right, we have (a) a Gaussian kernel function, (b) the Gaussian function subject to intensity based constraint with $\epsilon = 1/3$, (c) the Gaussian function subject to intensity based constraint with $\epsilon = 10^{-2}$. It is clear that the base of the function is modified ever for small ϵ.



(a) Intensity Based Constraint (b) Location Based Constraint

Figure 4.19: Illustration of the effect of location based constraints on a Gaussian function. From left to right, we have (a) the Gaussian function subject to intensity based constraint with $\epsilon = 10^{-2}$, (b) the Gaussian function subject to location based constraint with $\delta = 15$. The location based constraint allows the blur function to retain its structure while the intensity based constraint does not.

4.3.9 Solution Algorithm

In order to solve this model we minimise (4.32) alternately with respect to the blur function $h(\boldsymbol{x})$ and the image $u(\boldsymbol{x})$. At each iteration, we implement the constraints on the blur function given by (4.35) and (4.30) and the projection constraint on the image to the non-negative range given by (4.31). We present the solution algorithm for this model in Algorithm 13 where we use the notation ℓ_b^a to denote the step $\ell - 1 + a/b$.

Algorithm 13 Deblurring with Global Support Kernels (DGSK)

function $(h(\boldsymbol{x}), u(\boldsymbol{x})) \leftarrow$DGSK$(z(\boldsymbol{x}), h_{initial}(\boldsymbol{x}), \alpha_1, \alpha_2, tol, maxit)$
  $u^{(0)}(\boldsymbol{x}) \leftarrow z(\boldsymbol{x})$
  $h^{(0)}(\boldsymbol{x}) \leftarrow h_{initial}(\boldsymbol{x})$
  for $\ell \leftarrow 1$ **to** $maxit$ **do**
    Solve for $h^{(\ell_4^1)}(\boldsymbol{x})$:

$$u^{(\ell-1)}(-\boldsymbol{x}) * \left(u^{(\ell-1)}(\boldsymbol{x}) * h^{(\ell_4^1)}(\boldsymbol{x}) - z(\boldsymbol{x})\right) - \alpha_2 \nabla^2 h^{(\ell_4^1)}(\boldsymbol{x}) = 0$$

    Impose: $h^{(\ell_4^2)}(\boldsymbol{x}) \leftarrow \mathbb{P}^G\left(h^{(\ell_4^1)}(\boldsymbol{x})\right)$
    Impose: $h^{(\ell_4^3)}(\boldsymbol{x}) \leftarrow \mathcal{C}_1\left(h^{(\ell_4^2)}(\boldsymbol{x})\right)$
    Impose: $h^{(\ell)}(\boldsymbol{x}) \leftarrow \mathcal{C}_2\left(h^{(\ell_4^3)}(\boldsymbol{x})\right)$
    Solve for $u^{(\ell_2^1)}(\boldsymbol{x})$:

$$h^{(\ell)}(-\boldsymbol{x}) * \left(h^{(\ell)}(\boldsymbol{x}) * u^{(\ell_2^1)}(\boldsymbol{x}) - z(\boldsymbol{x})\right) - \alpha_1 \nabla \cdot \left(\frac{\nabla u^{(\ell_2^1)}(\boldsymbol{x})}{|\nabla u^{(\ell_2^1)}(\boldsymbol{x})|_\beta}\right) = 0$$

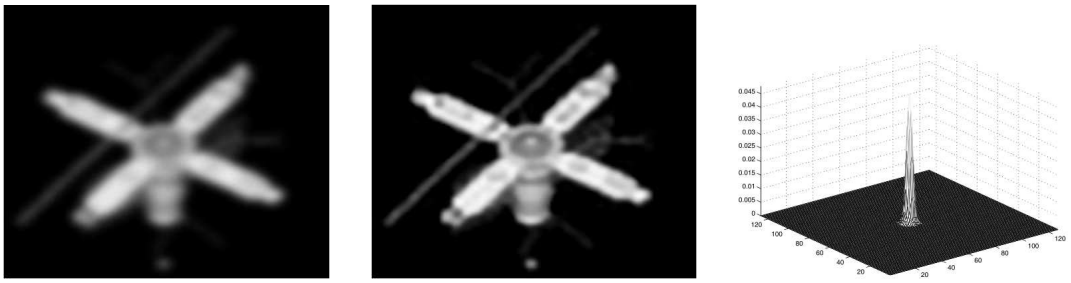
    Impose: $u^{(\ell)}(\boldsymbol{x}) \leftarrow \mathbb{P}\left(u^{(\ell_2^1)}(\boldsymbol{x})\right)$
  end for
  $h(\boldsymbol{x}) \leftarrow h^{(\ell)}(\boldsymbol{x})$
  $u(\boldsymbol{x}) \leftarrow u^{(\ell)}(\boldsymbol{x})$
end function

4.3.10 Experimental Results

We include results of restoring by Algorithm 13 satellite and retina images corrupted by Gaussian blur in Figures 4.20—4.21. In both cases, the image is restored, with the psnr of the satellite image increasing by 4.31dB and that of the retina image increasing by 5.46dB.

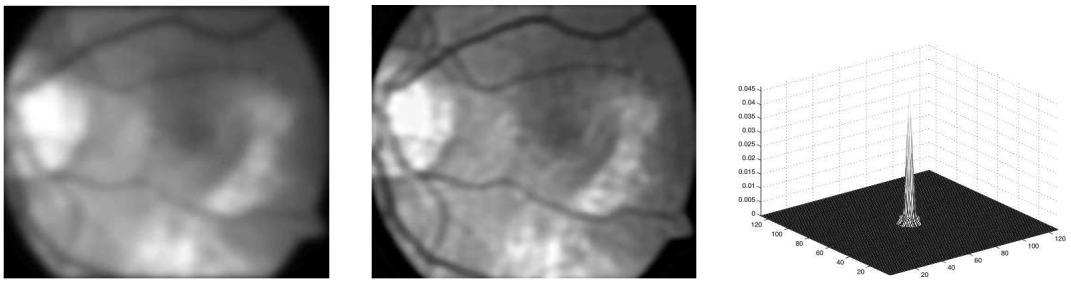
4.3.11 Conclusion

In this section, we have presented modified blur function constraints for blind deconvolution which restrict the possible approximations of the blur function. In this way, we have obtained improved results when restoring images corrupted by Gaussian blur as



(a) Received Image (b) Restored Image (c) Approximated Blur Function

Figure 4.20: Restoration of the blurred satellite image by Algorithm 13. From left to right, we have (a) the received image, (b) the restored image, (c) the approximated blur function. The restored image of psnr 24.71 shows visible improvement on the received image which has psnr 20.40.



(a) Received Image (b) Restored Image (c) Approximated Blur Function

Figure 4.21: Restoration of the blurred retina image by Algorithm 13. From left to right, we have (a) the received image, (b) the restored image, (c) the approximated blur function. The restored image of psnr 30.38 shows visible improvement on the received image which has psnr 24.92.

well as local support blur such as out of focus. We have also presented a mixed regularisation model employing the H^1 semi-norm in the case of regularisation for Gaussian kernel functions.

While these new optimisation constraints are capable of obtaining good results, they are not generally reliable since they require prior knowledge of the type of blur and an appropriate choice of the ϵ or δ values. However, it should be noted that these parameters can be chosen to allow a larger range of blur functions by placing less of a restriction on the range.

It can also be noted that, along with the other constraints which we apply, we are making a significant change to the blur function at each iteration which may prevent the final restored blur function from being a minimiser of the relevant functional together with the restored image. We consider in later chapters the implicit application of constraints, aiming to avoid the necessity for significantly altering the blur function in this way at each iteration.

Chapter 5

A New Constrained Deblurring Model

We have so far considered improvements to deblurring models introduced in Chapter 3 by considering some modifications in Chapter 4 which enable us to achieve lower levels of error in less cpu time. In this chapter, we present the first piece of our main work which aims to obtain improved results in the case of restoring images from known blur degradation by applying constraints on the image intensity values implicitly.

Although image intensities are non-negative quantities, imposing positivity is not always considered in restoration models due to a lack of simple and robust methods of imposing the constraint. In this chapter, we propose a suitable exponential type transform and apply it to the commonly-used total variation model, introduced in §3.3.5, to achieve an implicitly constrained solution (positivity at its lower bound and a prescribed intensity value at the upper bound). Numerical algorithms are presented to solve the resulting non-linear partial differential equations. Test results show that the proposed method is competitive when compared with existing methods in simple cases and more superior in other cases.

5.1 Introduction

Deconvolution in the case of *known* blur, has been investigated widely in the last few decades giving rise to a variety of solutions [9, 11, 44, 94, 119, 104, 178, 177, 180, 182, 181, 176]. In non-blind deconvolution, the point spread function is assumed known even though this information is not available in most of the real applications. In many cases, we know that our restored image must have strictly non-negative intensities, but the solution by traditional methods may yield results which are not necessarily positive. This has implications for most images with significant amounts of dark space, i.e. images with many pixel intensity values close to or equal to zero, as well as for blind deconvolution where the representation of certain blur functions has a significant amount of zero or near-zero values.

In this chapter, we present a model for non-blind deconvolution which not only en-

sures a strictly positive result but also limits the upper boundary of the image intensity values, keeping them within a prescribed range. Related work in this area can be found as early as [19] and more work has been carried out in recent years which attempts to find strictly positive solutions for several applications, particularly astronomical imaging. Vogel and Bardsley [13] gave a method for large-scale minimisation problems with non-negativity constraints using a cost functional including the statistics of the noise in the image data. A reduced Newton method was introduced such that Newton steps are only taken in the inactive variables, meaning those which are non-zero. A sparse matrix preconditioner was also introduced to improve convergence of Conjugate Gradient which is used to compute approximate reduced Newton steps. Benvenuto et al. [15] attempted to increase the efficiency of the projected Langweber method and iterative image space reconstruction algorithm, both of which demonstrate the property of semi-convergence. The results of the algorithms improve at the earlier iterations and then begin to worsen. The algorithms are also quite slow. The aim of Benvenuto et al. was primarily to improve the speed and convergence of these algorithms. The works of [34, 35] proposed other ideas based on non-negative projections for deblurring. More recently, Chan et al. [44] gave a method for constrained image deblurring which is related to [13] but uses efficient alternate direction methods to drive the restored image closer to a projection of itself onto the ideal range. Since such projections (typically scaling or truncation) may cause a decrease in quality if simply applied at the end, the authors of [44] improve results by successively forcing the intensity values of the image to lie within a range which tends towards the ideal.

The rest of the chapter is organised as follows. §5.2 briefly reviews the total variation based variational models for deblurring introduced in §3.3.5. §5.3 presents our proposed transform and consequently its resulting model and algorithms. §5.4 discusses some refinement issues followed by §5.5 of numerical results and §5.6 of conclusions.

5.2 The Total Variation based Deblurring Models

Noise and blur can be commonly found in digital images due to factors such as imperfections of the capturing equipment and scattering through non-homogeneous medium. Following the work of [169], we consider the linear deblurring problem with additive noise

$$z(\boldsymbol{x}) = [k * u](\boldsymbol{x}) + \eta(\boldsymbol{x}), \qquad \boldsymbol{x} = (x, y) \in \Omega \tag{5.1}$$

where z is the (known) observed image, k is the point spread function, u is the image we aim to approximate and η is the unknown noise function. There are two related models that one may consider.

Given knowledge of the blurring kernel k, the total variation regularised model

[169, 47, 13, 193] reconstructs u from solving the minimisation problem

$$\min_u \int_\Omega (k * u - z)^2 \, \mathrm{d}\Omega + \alpha_1 \|u\|_{TV}^\beta, \qquad \|u\|_{TV}^\beta = \int_\Omega |\nabla u|_\beta, \tag{5.2}$$

introduced in §3.3.5 where $\alpha_1 > 0$ and $\|u\|_{TV}^\beta = \int_\Omega |\nabla u|_\beta = \int_\Omega \sqrt{u_x^2 + u_y^2 + \beta}$ is a smooth approximation of the total variation, where β is a small non-negative constant. The model has been widely studied. In addition to the regularisation parameter selection for variational denoising considered in §2.5.4, algorithms for optimal selection of the parameter α_1 have also been proposed for deblurring problems in [70, 205, 59].

We can also consider the related problem of restoring a blur function given the true image. Then to restore the kernel k from a known image u, a related model to (5.2) may be proposed

$$\min_k \int_\Omega (u * k - z)^2 \, \mathrm{d}\Omega + \alpha_2 \|k\|_{TV}^\beta, \quad \text{s. t. } \; k \geq 0, \; \int_\Omega k(s,t) \, \mathrm{d}s \, \mathrm{d}t = 1, \tag{5.3}$$

where $\alpha_2 > 0$ and we have used the equality $u * k = k * u$. Note that we have imposed the additional constraints of non-negativity and unit integral which we may assume to be the appropriate for a blur function. Our main concern in this chapter is equation (5.2).

Here we remark that for (5.2), from our experience, the positivity method from [193] appears to be reliable. However, for model (5.3), the method of projecting solutions to satisfy the constraints $k \geq 0$, $\int_\Omega k(s,t) \, \mathrm{d}s \, \mathrm{d}t = 1$ seems less robust. Therefore, it is of importance to seek alternative and effective methods.

5.3 A Transform Based Method for Implicitly Constrained Reconstruction

In this section, we present a new transform method for imposing positivity for solving models (5.2)-(5.3). Our method will transform our constrained model to an unconstrained one. Therefore the positivity constraint is automatically satisfied. Below we use model (5.2) as the example.

Our motivation comes from a simple idea. If we wish for $u \geq 0$, we set $u = \exp(\psi)$ and reformulate our model in the new variable ψ. Then for any ψ, we can ensure $u \geq 0$. However, this seemingly great idea does not work because the inverse transform $\psi = \ln u$ does not allow $u = 0$. A remedial solution is to define the modified transform $u = \exp(\psi) - \epsilon$ so $\psi = \log(u + \epsilon)$; however to ensure $u \geq 0$, we require $\psi \geq \log(\epsilon)$ which implies that ψ must be constrained i.e. the underlying transform is not suitable. We would therefore aim to choose ϵ to be a very small positive number so that any final projection, if necessary, would have minimal effect on the result.

In order to impose a constraint on both the upper and lower bounds of u, we have

found that a suitable exponential type transform is the following

$$u = \tilde{H}_\epsilon(\psi) = \frac{w + 2b}{1 + e^{-\frac{\psi}{\epsilon}}} - b$$

which resembles a smooth approximation to the Heaviside function given by

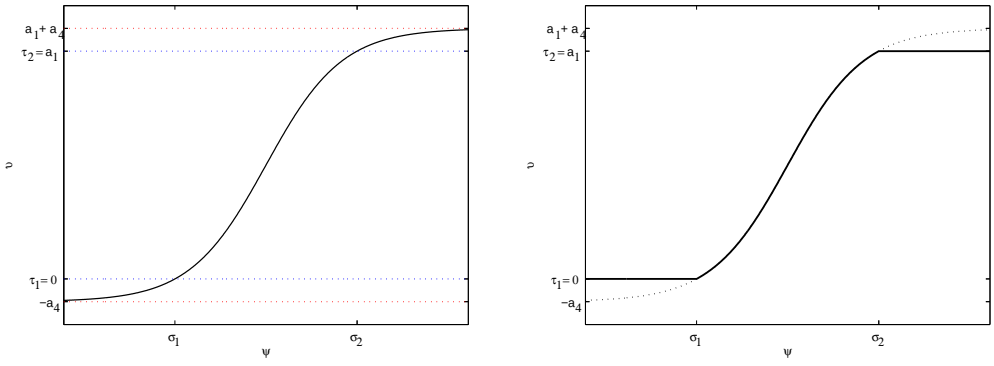
$$H(\psi) = \begin{cases} 0, & \text{if } \psi < 0 \\ 1, & \text{if } \psi \geq 0, \end{cases}$$

where $\epsilon, b, w > 0$, and $0 \approx -b \leq \tilde{H}_\epsilon(\psi) \leq w + b \approx w$ defines the intensity range for *any* ψ. Practically one may take, for (5.3), $b = 0.1, w = 255$ to accommodate the commonly used range $u \in [0, 255]$ and, for (5.27), $b = 0.01, w = 1$ to allow $k \in [0, 1]$. Note the inverse transform $\psi = -\frac{\epsilon}{2} \log \frac{w-u}{u+b}$ allows $u = 0$.

To allow generality, our proposed transform will be of the form

$$T(\psi) = \frac{a_1 + 2a_4}{1 + a_2 e^{\frac{-2\psi}{a_3}}} - a_4 \tag{5.4}$$

where $\boldsymbol{a} = (a_1, a_2, a_3, a_4)$ and all a_j's are positive. Note $0 \approx -a_4 \leq u = T(\psi) \leq a_1 + a_4 \approx a_1$ for any ψ. As illustrated in Figure 5.1, the generality allows us to adjust the maximal and minimal values of the range using a_1 and a_4, the spread of usable range of ψ using a_3 and the point of u at which ψ will be equal to zero using a_2. We can, if we wish, use this to restrict all values of ψ to positive but this is not necessary. We consider the choice of parameters in more detail in Appendix 5.A.



(a) Transform of ψ (b) Projected transform

Figure 5.1: Graph of Heaviside Transform $u = T(\psi)$. On the left, we have (a) the transform of ψ with an exaggerated a_4 and on the right (b) the projected transform onto the range $[\tau_1, \tau_2]$.

Once the transform is specified, we now consider how to use it to reconstruct ψ first and hence the image u. The model (5.2) as studied in [193] can be transformed from

$$\min_u f(u) = \frac{1}{2} \left\| \int k(x - x', y - y') u(x', y') \, \mathrm{d}\Omega - z(x, y) \right\|^2_{L^2(\Omega)} + \alpha L(u)$$

(with $u \geq 0$) to the new problem for ψ

$$\min_{\psi} f(\psi) = \frac{1}{2}\Big\| \int k(x-x', y-y')T(\psi(x',y'))\,\mathrm{d}\Omega - z(x,y)\Big\|^2_{L^2(\Omega)} + \alpha L(T(\psi)) \quad (5.5)$$

where L denotes the TV regulariser for (5.2) and the H_1 for (5.9). The new and transformed model (5.5) has no constraint on ψ and yet can ensure (5.3) to have a positive solution u. However, since both terms in (5.5) are non-linear in ψ, it remains to address the numerical solution methods.

In what follows, we shall propose to treat term 1 in (5.5) by linearising $T(\psi)$ (due to the challenge associated with a non-local operator k) and term 2 by lagged diffusion ideas (as for solving the denoising [193]).

Linearisation of $T(\psi)$

The Taylor expansion of $T(\psi)$ about $\psi = 0$ is given by

$$T(\psi) = A + B\psi + O\left(\psi^2\right), \quad A = \frac{a_1 + 2a_4}{1+a_2} - a_4,\ B = \frac{2a_2(a_1+2a_4)}{(1+a_2)^2 a_3}.$$

Thus we can decompose $T(\psi)$ by separating its linear term in the form

$$u = T(\psi) = A + B\psi \ + \ \bar{\bar{v}}(\tilde{\psi}), \quad \bar{\bar{v}}(\tilde{\psi}) = \bar{v}(\tilde{\psi}) - A,\ \bar{v}(\tilde{\psi}) = T(\tilde{\psi}) - B\tilde{\psi}.$$

Iterative minimisation

Using the above decomposition, our solution strategy is as follows:

1: $u^{(0)} \leftarrow z$ and $\psi^{(0)} \leftarrow T^{-1}(u^{(0)})$
2: **for** Iterate on ℓ **do**
3: Solve for $\psi^{(\ell+1)}$, given $\psi^{(\ell)}$, from

$$\psi^{(\ell+1)} \leftarrow \min\left\{ \|k * \psi^{(\ell+1)} B - \bar{z}(\psi^{(\ell)})\|^2_{L^2(\Omega)} + \alpha \|B\psi^{(\ell+1)}) + \bar{v}(\psi^{(\ell)})\|^{\beta}_{TV} \right\} \quad (*)$$

4: where $\bar{z}(\psi^{(\ell)}) = z - k * \bar{v}(\psi^{(\ell)})$.
5: **end for**

We now discuss how to solve the above equation (*) i.e.

$$\min_{\psi} \left\{ f(\psi) = \frac{1}{2}||Bk * \psi - \bar{z}||^2_{L^2(\Omega)} + \alpha \int_{\Omega} |\nabla\left(B\psi + \bar{v}\right)|_{\beta}\,\mathrm{d}\Omega \right\}. \quad (5.6)$$

Consider each term in turn. First let $f_1 = \frac{1}{2}||Bk * \psi - \bar{z}||^2_{L^2(\Omega)}$ so $\min_{\psi} f_1$ is given when $\partial f_1/\partial\psi = 0$. Here

$$\begin{aligned} \frac{\partial f_1}{\partial \psi} &= \frac{\partial}{\partial \psi}\frac{1}{2}||Bk * \psi - \bar{z}||^2_{L^2(\Omega)} \quad = \frac{1}{2}\frac{\partial}{\partial\psi}(Bk * \psi - \bar{z})^2 \\ &= \left(\frac{\partial}{\partial\psi}(Bk * \psi)\right)(Bk * \psi - \bar{z}) \quad = (Bk)^T(Bk * \psi - \bar{z}). \end{aligned}$$

Second let $f_2 = \int_\Omega \nabla(B\psi + \bar{v})\,\mathrm{d}\Omega$ and $\min_\psi f_2$ is given when $\dfrac{\partial}{\partial\epsilon}(f_2(\psi+\epsilon\phi))|_{\epsilon\to 0} = 0$ for an arbitrary function ϕ. We have

$$\begin{aligned}
\frac{\partial}{\partial\epsilon} f_2(\psi+\epsilon\phi)\bigg|_{\epsilon\to 0} &= \frac{\partial}{\partial\epsilon}\int_\Omega |\nabla(B(\psi+\epsilon\phi)+\bar{v})|_\beta\,\mathrm{d}\Omega\bigg|_{\epsilon\to 0} = \int_\Omega \frac{\partial}{\partial\epsilon}\,|\nabla(B(\psi+\epsilon\phi)+\bar{v})|_\beta\,\mathrm{d}\Omega\bigg|_{\epsilon\to 0} \\
&= \int_\Omega \frac{\nabla(B(\psi+\epsilon\phi)+\bar{v})}{|\nabla(B(\psi+\epsilon\phi)+\bar{v})|_\beta}\cdot\nabla B\phi\,\mathrm{d}\Omega\bigg|_{\epsilon\to 0} = \int_\Omega \frac{\nabla(B\psi+\bar{v})}{|\nabla(B\psi+\bar{v})|_\beta}\cdot\nabla B\phi\,\mathrm{d}\Omega \\
&= -\int_\Omega \nabla\cdot\left(\frac{\nabla(B\psi+\bar{v})}{|\nabla(B\psi+\bar{v})|_\beta}\right) B\phi\,\mathrm{d}\Omega + \int_\Gamma \frac{\nabla(B\psi+\bar{v})}{|\nabla(B\psi+\bar{v})|_\beta}\cdot B\phi\vec{n}\,\mathrm{d}\Gamma,
\end{aligned}$$

where $|\nabla(\cdot)|_\beta$ is given in §5.2. We have therefore that $\min_\psi\{f = f_1 + f_2\}$ is solved by

$$(Bk)^T(Bk*\psi - \bar{z}) + \alpha\nabla\cdot\left(\frac{\nabla(B\psi+\bar{v})}{|\nabla(B\psi+\bar{v})|_\beta}\right) B = 0 \tag{5.7}$$

where $\bar{z} = \bar{z}(\psi) = z - k*\bar{v}(\psi)$ and $\bar{v} = \bar{v}(\psi) = T(\psi) - B\psi$.

Overall Algorithm

Assume u has a Dirichlet boundary condition. Then the discretised the Point Spread Function (PSF) k leads to a Block Toeplitz matrix with Toeplitz Blocks (BTTB) [94, 193]. In order to define the transform, we calculate the parameters $a_1,\ldots,a_4$ according to the Appendix. We calculate the initial estimate of $\psi^{(0)}$ given the initial estimate of $u^{(0)}$ as follows:

$$u = T(\psi) = \frac{a_1+2a_4}{1+a_2e^{-\frac{2\psi}{a_3}}} - a_4, \quad \psi = T^{-1}(u) = -\frac{a_3}{2}\ln\left(\frac{a_1-u+a_4}{a_2(u+a_4)}\right).$$

We then solve the Euler Lagrange equation (5.7) and finally transform the image back, obtaining our restored image u with positive entries. This is shown in Algorithm 14.

Algorithm 14 A Transform Based Constrained Deblurring Algorithm (TCD)

1: **function** $u \leftarrow$TCD$(z,k,\alpha,\beta,\boldsymbol{a},tol,maxit)$
2: $u^{(0)} \leftarrow z$
3: Calculate $\boldsymbol{a} = \{a_1,a_2,a_3,a_4\}$
4: $\psi^{(0)} \leftarrow -(a_3/2)\log\left(\left(a_1+a_4-u^{(0)}\right)/\left(a_2(u^{(0)}+a_4)\right)\right)$
5: **for** $\ell \leftarrow 1$ **to** $maxit$ **do**
6: Solve equation for $\psi^{(\ell+1)}$ given $\psi^{(\ell)}$, i.e.

$$\psi^{(\ell+1)} \leftarrow \text{ SOLVE } (Bk)^T * \left(k*\psi^{(\ell+1)} - \overline{z}(\psi^{(\ell)})\right) - \alpha\nabla\cdot\frac{\nabla\left(B\psi^{(\ell+1)} - \overline{v}^{(\ell)}\right)}{\left|\nabla\left(B\widehat{\psi}^{(\ell+1)} - \overline{v}^{(\ell)}\right)\right|_\beta} = 0$$

7: where $\overline{z}(\psi^{(\ell)}) = z - k*\overline{v}(\psi^{(\ell)})$ and $\widehat{\psi}$ denotes a lagging from ψ.
8: **end for**
9: On exit, $u \leftarrow (a_1+2a_4)/\left(1+a_2\exp\left(-2\psi^{(\ell+1)}\right)/a_3\right)$.
10: **end function**

5.4 Refinements and other Solution Strategies

In this section, we present some refinements to the model presented in the previous section by alternative linearisation and regularisation. We also consider obtaining a closer approximation of the image for the initial estimate of this model and improving the cpu time with an accelerated model. Finally, we present a convex model obtained by a small relaxation on the functional.

5.4.1 Alternative Linearisation

In order to improve the speed of obtaining a solution, we carry out the Total Variation norm linearisation alongside the updating of the linearisation of the transform, thereby solving

$$(Bk)^T * \left(k * \psi^{(\ell+1)} - \bar{z}(\psi^{\ell})\right) - \alpha\nabla \cdot \frac{\nabla\left(B\psi^{(\ell+1)} - \bar{v}(\psi^{\ell})\right)}{\left|\nabla\left(B\psi^{(\ell)} - \bar{v}(\psi^{\ell})\right)\right|_{\beta}} = 0. \tag{5.8}$$

In this way, we hope to get speed-up due to the saving of iterations on $\widehat{\psi}$. Experimental results are shown in Figure 5.8 and error values and cpu times for this method and the previous transform method are given in Table 5.6. It can be noted that, the reduction in cpu time is significant.

5.4.2 Alternative Regularisation

While the total variation semi-norm which we have used in our model gives good results for images which have sharp changes in intensity and hence jumps in the pixel intensity value, improved results may be found by considering alternative regularisation to treat smooth images. In this section, we consider a simple form of alternative regularisation using the L^2-norm of the gradient of the image. More robust regularisations are based on high order regularisers; see [26, 23, 153, 56].

In the traditional case, using a least squares fitting term and L^2 as a regularisation term, we will obtain a linear partial differential equation to solve. We give this minimising functional as

$$f(u) = \frac{1}{2}||k * u - z||^2_{L^2(\Omega)} + \frac{\alpha}{2}\int |\nabla u|^2 \, \mathrm{d}\Omega. \tag{5.9}$$

The well-known Euler-Lagrange equation for the image u is therefore given by

$$k^T * (k * u - z) - \alpha\Delta u = 0. \tag{5.10}$$

Now referring to the above section, we substitute $u = B\psi + \bar{v}\left(\tilde{\psi}\right)$ to (5.9)

$$\begin{aligned} f(u) &= \frac{1}{2}||k * (B\psi + \bar{v}\left(\tilde{\psi}\right)) - z||^2_{L^2(\Omega)} - \alpha\int |\nabla\left(B\psi + \bar{v}\left(\tilde{\psi}\right)\right)|^2 \, \mathrm{d}\Omega \quad (5.11) \\ &= \frac{1}{2}||Bk * \psi - \bar{z}(\tilde{\psi})||^2_{L^2(\Omega)} - \alpha\int |\nabla\left(B\psi + \bar{v}\left(\tilde{\psi}\right)\right)|^2 \, \mathrm{d}\Omega \quad (5.12) \end{aligned}$$

where $\bar{z}(\psi) = z - k * \bar{v}\left(\tilde{\psi}\right)$ and $\bar{v}\left(\tilde{\psi}\right) = T\left(\tilde{\psi}\right) - B\tilde{\psi}$. The linearised Euler-Lagrange equation is

$$k^T * \left(Bk * \psi - \bar{z}\left(\tilde{\psi}\right)\right) - \alpha\Delta\left(B\psi + \bar{v}\left(\tilde{\psi}\right)\right) = 0. \tag{5.13}$$

5.4.3 Initialisation of u and k

Since there exist many efficient algorithms for solving models (5.2) and (5.3) without the positivity constraints, one idea of acquiring good initialisations for u and k is through applying such algorithms first.

In fact, the simplistic L^2 method given by minimising (5.9) leads to solving the linear partial differential equation (5.10) which can be done efficiently. We may therefore use the solution of it as the initial estimate u and then our transform model will offer a positive solution.

As we shall see from the next section, for model (5.3) with the unknown kernel k, the Vogel's method [193] is no longer effective but we may use its result as an initial guess for our transform model; see Table 5.7 and Figure 5.9.

5.4.4 An Acceleration Algorithm for the Model

While our model performs well, it can often be rather slow to execute, particularly in cases of Gaussian blur. We address this issue using an alternating direction method (ADM) [44, 104, 198, 197]. We aim to separate our model into one of deblurring and one of denoising, each of which can be executed reasonable quickly. Starting with the unconstrained non-negative functional given by equation (5.5) we use the ADM to create the augmented Lagrangian functional

$$f(u,\psi,\lambda) = \frac{1}{2}||k*u-z||^2_{L^2(\Omega)} + \alpha L(T_{\boldsymbol{a}}(\psi)) + \frac{\gamma}{2}||u - T_{\boldsymbol{a}}(\psi)||^2_{L^2(\Omega)} + <\lambda, u - T_{\boldsymbol{a}}(\psi)> \tag{5.14}$$

where L represents either total variation (where we expect jumps in intensity) or L^2 (where we expect smooth edges) i.e.

$$L(u) = \int_\Omega |\nabla u|_\beta, \qquad \text{or} \qquad L(u) = \int_\Omega |\nabla u|^2 \, \mathrm{d}\Omega. \tag{5.15}$$

Our aim is now to minimise f with respect to u, ψ and λ. Then we can give the Euler Lagrange equation for u:

$$k^T * (k * u - z) + \gamma\left(u - T_{\boldsymbol{a}}(\psi)\right) + \lambda = 0 \tag{5.16}$$

and, rearranging, we have

$$\left(k^T * k + \gamma\delta\right) * u = k^T * z + \gamma T_{\boldsymbol{a}}(\psi) - \lambda \tag{5.17}$$

where δ denotes the delta function and we can solve this using Fourier transforms. For

additional support, we might add a term for u, given by $\chi L_1(u)$ where $\chi > 0$ and L_1 is a regularisation term. This model can be achieved by setting $\chi = 0$.

For the second equation, we minimise with respect to ψ as follows. We must deal with the non-linearity of the transform. We do this by considering the Taylor expansion given by

$$T_{\boldsymbol{a}}(\psi) = A + B\psi + O(\psi^2)$$

and approximate the transform with $T_{\boldsymbol{a}}(\psi) = B\psi + R(\psi)$ where R, the residual, is given by $R = T_{\boldsymbol{a}}(\psi) - B\psi$. In practice, we will use this to form a fixed-point lagging technique by substituting $T_{\boldsymbol{a}}(\psi, \tilde{\psi}) = B\psi + R(\tilde{\psi})$, lagging $\tilde{\psi}$ and updating until $||\psi - \tilde{\psi}||$ is sufficiently small.

$$-B\lambda - \gamma B\left(u - (B\psi + \tilde{R})\right) + \alpha L(\tilde{\psi})\psi = 0 \tag{5.18}$$

where, for total variation, we have[1]

$$L(\tilde{\psi})\psi = \frac{4\tilde{E}_1(a_1 + 2a_4)(\tilde{E}_1 - 1)|\nabla\psi|_\beta}{(1 + \tilde{E}_1)^3 a_3^2} - \nabla \cdot \left(\frac{2(a_1 + 2a_4)\tilde{E}_1}{(1 + \tilde{E}_1)^2 a_3 |\nabla\tilde{\psi}|_\beta}\nabla\psi\right).$$

Overall Algorithm

In order to solve our model, we begin with the initial estimate (typically the received image) and calculate the initial estimate of ψ using the chosen parameters. We then proceed to solve for u and ψ, updating λ. Our algorithm is given below in Algorithm 15.

[1] See Appendix 5.B for details.

Algorithm 15 Accelerated Transform Based Constrained Deblurring (ATCD)

function $u \leftarrow$ATCD$(z, k, \alpha, \beta, \gamma, \lambda^{(1)}, \boldsymbol{a}, tol, maxit)$
  $u^{(0)} \leftarrow z$
  Calculate $\boldsymbol{a} = \{a_1, a_2, a_3, a_4\}$
  $\psi^{(0)} \leftarrow -(a_3/2)\log\left(\left(a_1 + a_4 - u^{(0)}\right) / \left(a_2(u^{(0)} + a_4)\right)\right)$
  for $\ell_1 \leftarrow 1$ **to** $maxit$ **do**
    for $\ell_2 \leftarrow 1$ **to** $maxit$ **do**
      Solve equation for $u^{(\ell_2+1)}$ given $u^{(\ell_2)}$, i.e.

$$u^{(\ell_2+1)} \leftarrow \text{ SOLVE } k^T(ku^{(\ell_2+1)} - z) + \gamma(u^{(\ell_2+1)} - (B\psi + \tilde{R})) + \chi L_1(\tilde{u^{(\ell_2)}})u^{(\ell_2+1)} = -\lambda^{(\ell_1)}$$

    end for
    for $\ell_3 \leftarrow 1$ **to** $maxit$ **do**
      Solve equation for $\psi^{(\ell_3+1)}$ given $\psi^{(\ell_3)}$, i.e.

$$\psi^{(\ell_3+1)} \leftarrow \text{ SOLVE } \quad -\gamma B\left(u^{(\ell_2+1)} - (B\psi^{(\ell_3+1)} + \tilde{R})\right) + \alpha L(\tilde{\psi^{(\ell_3)}})\psi^{(\ell_3+1)} = B\lambda^{(\ell_1)}$$

    end for
    Update $\lambda^{(\ell_1+1)} \leftarrow \lambda^{(\ell_1)} + \gamma\left(u^{(\ell_2+1)} - T_{\boldsymbol{a}}\left(\psi^{(\ell_3+1)}\right)\right)$.
  end for
  On exit, $u \leftarrow (a_1 + 2a_4)/(1 + a_2\exp(-2\psi^{(\ell_3+1)}/a_3))$.
end function

5.4.5 A Convex Accelerated Model

We now wish to prove convergence of Algorithm 15. However, due to the lack of convexity of the model (5.14), this is not trivial. We therefore propose below a relaxation of this model so that the new model is convex by the addition of a suitable term. We can then show convergence from the established approaches (see [100, 193, 145]). Tests in §5.5 will demonstrate that such a relaxation does not have a considerable impact on the solution or the quality of the restoration.

We aim to find an appropriate convex relaxation of this model by considering the fitting and regularisation terms separately since the sum of two convex functions is also convex. We attempt to obtain convexity of the fitting terms with the addition of a fitting term involving the function ψ of the form

$$\mu \int_\Omega (\psi - \zeta)^2 \, \mathrm{d}\Omega$$

where ζ is a function not depending on ψ and μ is a non-negative real constant which must be sufficiently large to make the model (5.14) convex. In fact we see that, for this model, μ may be quite small so that assuming close proximity of the arguments this term should have only a small impact on the results. ζ should be a function which is approximately equal to ψ but not depend on u so that convexity with respect to u is unaffected. We let $\zeta = T_{\boldsymbol{a}}^{-1}\left(\mathbb{P}(z^*)\right)$ where z^* is a solution given by a model which can be solved efficiently and $\mathbb{P} : \mathbb{R}^{m\times n} \to [0, a_1]^{m\times n}$ denotes a projection onto the required

range by truncation which is required for the function $T_{\boldsymbol{a}}^{-1}$. In this case, we use

$$z^* = \min_u \left\{ \int_\Omega (k * u - z)^2 \, \mathrm{d}\Omega + \alpha \int_\Omega |\nabla u|^2 \, \mathrm{d}\Omega \right\}$$

which can be found by solving the equation

$$(k^T * k - \alpha D) * u = k^T * z$$

directly where D is a term describing the finite difference approximation of the second derivative using convolution.

With similar considerations for the regularisation term, we present our modified convex functional as

$$f(u, \psi; \lambda) = \frac{1}{2}||k * u - z||^2_{L^2(\Omega)} + \frac{\gamma}{2}||T_{\boldsymbol{a}}(\psi) - u||^2_{L^2(\Omega)} + < \lambda, T_{\boldsymbol{a}}(\psi) - u >$$

$$+\mu||\psi - \zeta||^2_{L^2(\Omega)} + \alpha \int_\Omega \left| \nabla \left(T_{\boldsymbol{a}}(\psi) + \theta||\psi - \zeta||^2_{L^2(\Omega)} \right) \right|_\beta \tag{5.19}$$

where μ and θ must satisfy

$$\mu \geq \frac{8\,(a_1 + 2a_4)}{27 a_3^2} \left(2\gamma(a_4 + \mathcal{L}_u) + \mathcal{L}_\lambda\right), \qquad \theta \geq -\frac{2(a_1 + 2a_4)(3\sqrt{3} - 5)}{\left(3 - \sqrt{3}\right)^3 a_3^2}. \tag{5.20}$$

To give an example of the values for the parameters, if we assume that our image is contained in the range $[0, 1]$, $\mathcal{L}_u = \mathcal{L}_\lambda = 0$, then we may take $\boldsymbol{a} = \{1, 1, 0.44, 0.01\}$, $\mu \geq 0.04\gamma$ and $\theta \geq -1$. In this case, assuming that γ is not chosen to be high, it is not essential that ζ be a very close approximation of ψ. That is, making use of the solution of a simple model to obtain the approximation z^* should not have a significant impact on the quality of the final result.

In order to minimise the functional, we first calculate ζ and proceed with alternate minimisation. We present our overall algorithm below in Algorithm 16. For brevity, we do not present the Euler-Lagrange equation for ψ but it can be calculated in a similar manner to those above.

Algorithm 16 Convex Transform based Implicitly Constrained Deblurring (CTCD)

function $u \leftarrow$CTCD$(h, z, \lambda^{(1)}; \alpha_s, \gamma, \alpha, \mu, \theta)$
  Solve equation for ζ, i.e.

$$z^* = \min_u \left\{ f_s(u) = ||h * u - z||^2_{L^2(\Omega)} + \alpha_s ||\nabla u||^2_{L^2(\Omega)} \right\}$$

  Calculate $\boldsymbol{a} = \{a_1, a_2, a_3, a_4\}$
  $u^{(0)} \leftarrow z$
  $\psi^{(0)} \leftarrow T_{\boldsymbol{a}}^{-1}(z)$
  $\zeta \leftarrow T_{\boldsymbol{a}}(\mathbb{P}(z^*))$
  for $\ell \leftarrow 1$ **to** *maxit* **do**
    Solve equation for $u^{(\ell+1)}$ given $u^{(\ell)}$, i.e.

$$u^{(\ell+1)} \leftarrow \text{ SOLVE } \left((h^\dagger h + \gamma\delta) * u^{(\ell+1)} = h^\dagger z + \lambda^{(\ell)} + \gamma \tau_{\boldsymbol{a}} \left(\psi^{(\ell)} \right) \right)$$

    Solve equation for $\psi^{(\ell+1)}$ given $\psi^{(\ell)}$, i.e.

$$\psi^{\ell+1} \leftarrow \min_\psi \left\{ f\left(u^{(\ell+1)}, \psi; \lambda^{(\ell)} \right) \right\}$$

    Update $\lambda^{\ell+1} = \lambda^\ell + \gamma \left(u^{(\ell+1)} - T_{\boldsymbol{a}} \left(\psi^{(\ell+1)} \right) \right)$
  end for
  On exit, $u \leftarrow T_{\boldsymbol{a}} \left(\psi^{(\ell+1)} \right)$.
end function

We would now like to show that the functional defined above is convex.

Theorem 5.4.1 *Let $\Omega \subset \mathbb{R}^n$ be a non-empty convex subset of $\mathbb{R}^n$ and $f : \Omega \to \mathbb{R} \cup \{+\infty\}$ be the function defined by (5.19–5.20). Then f is convex with respect to the argument ψ for $\psi \in \Omega$.*

Proof. To show convexity, we show that the functional (5.19) is a sum of two convex functions. The first is given by

$$F(u, \psi) = \int_\Omega \gamma(\tau_{\boldsymbol{a}}(\psi) - u)^2 + \lambda(\tau_{\boldsymbol{a}}(\psi) - u) + \mu(\psi - \zeta)^2 \, \mathrm{d}\Omega. \tag{5.21}$$

where μ must satisfy the above constraint and the second is given by the regularisation term.

To demonstrate that equation (5.21) is convex, we require the second order derivative given by

$$\frac{\partial^2 F(u, \psi)}{\partial \psi^2} = 2\mu - 2j(\psi) \left[-2\gamma(\tau_{\boldsymbol{a}}(\psi) + a_4) a_2 E - (2\gamma(\tau_{\boldsymbol{a}}(\psi) - u) + \lambda)(a_2 E - 1) \right] \tag{5.22}$$

where $j(\psi) = \frac{2(\tau_{\boldsymbol{a}}(\psi) + a_4) a_2 E}{(1 + a_2 E)^2 a_3^2}$ and $E = E(\psi) := \exp(-2\psi/a_3)$ is non-negative.

It is not difficult to show that the term to the right of $j(\psi)$ is contained in the bound $(-\infty, 2\gamma(a_4 + \mathcal{L}_u) + \mathcal{L}_\lambda)$ where $\mathcal{L}_u$ and $\mathcal{L}_\lambda$ are the lower bounds of u and λ respectively.

For the function j, we can find that there is only one maximum by calculating the first derivative and finding the limits of the function as follows. We calculate the zero-point of the derivative

$$\frac{\partial j}{\partial \psi} = \frac{12(a_1+2a_4)a_2^2E^2}{(1+a_2E)^4a_3^3} - \frac{4(a_1+2a_4)a_2E}{(1+a_2E)^3a_3^3} = 0 \;\Leftrightarrow\; \psi = \frac{a_3}{2}\ln(2a_2)$$

at which the function j is non-negative and strictly positive assuming that at least one of a_1 and a_4 are non-zero, since $a_1, \ldots, a_4$ are non-negative constants.

Taking limits now and noting that $\lim_{\psi\to-\infty} E = \infty$ and $\lim_{\psi\to\infty} E = 0$, we find that the function j tends to 0 at $\pm\infty$ with a non-negative turning point given at $\psi = a_3\ln(2a_2)/2$ which must be the maximum.

$$\lim_{\psi\to-\infty} \frac{2\left(\frac{a_1+2a_4}{1+a_2E}\right)a_2E}{(1+a_2E)^2\,a_3^2} = \lim_{\psi\to-\infty} \frac{2\,(a_1+2a_4)\,a_2}{\left(\frac{1}{E}+3a_2+3a_2^2E+a_2^3E^2\right)a_3^2} = 0 \tag{5.23}$$

$$\lim_{\psi\to\infty} \frac{2\left(\frac{a_1+2a_4}{1+a_2E}\right)a_2E}{(1+a_2E)^2\,a_3^2} = 0. \tag{5.24}$$

Since the function tends to zero at both limits and has a single extremity, which is greater than or equal to zero, we can conclude that this is the maximum value and that the minimum is equal to zero, i.e.

$$j(\psi) \in \left(0, j\left(\frac{a_3}{2}\ln(2a_2)\right) = \frac{8\,(a_1+2a_4)}{27a_3^2}\right].$$

Substituting these bounds and inequalities into (5.22), it is clear that the convexity condition $\partial^2 F(u,\psi)/\partial\psi^2 \geq 0$ is satisfied.

For the total variation term, we begin by showing that if the function π is convex then its total variation is also convex. It will then remain to show that the function (5.26) is convex given the restriction on the value θ. We consider the duality definition [39]

$$J(\psi) = \sup\left\{-\int_\Omega \pi(\psi)\mathrm{div}\phi\ \mathrm{d}x : \phi \in C_c^\infty\left(\Omega;\mathbb{R}^N\right), |\phi(x)| \leq 1\,\forall x \in \Omega\right\}$$

where π is a function of ψ and we have $-\int_\Omega \pi(\psi)\mathrm{div}\phi\ \mathrm{d}x = \int_\Omega \phi\cdot\nabla\pi(\psi)\ \mathrm{d}x$. Letting $L_\phi : \psi \mapsto -\int_\Omega \pi(\psi)\mathrm{div}\phi\ \mathrm{d}x$, we would like to show that if π is convex then $J(\psi)$ is also convex. That is $\forall\ \psi_1, \psi_2$ and $t \in [0,1]$, we have $J(t\psi_1 + (1-t)\psi_2) \;\leq\; tJ(\psi_1) + (1-t)J(\psi_2)$. Assuming that $\pi(\psi)$ is convex with respect to ψ then we have the relation

$$\pi(t\psi_1 + (1-t)\psi_2) \;\leq\; t\pi(\psi_1) + (1-t)\pi(\psi_2)$$

and

$$L_\phi(t\psi_1 + (1-t)\psi_2) \;\leq\; tL_\phi(\psi_1) + (1-t)L_\phi(\psi_2) \;\leq\; tJ(\psi_1) + (1-t)J(\psi_2). \tag{5.25}$$

since J is the supremum of the functions L_ϕ. We have

$$\sup L_\phi(t\psi_1 + (1-t)\psi_2) = J(t\psi_1 + (1-t)\psi_2).$$

Consequently, we have by (5.25) that

$$J(t\psi_1 + (1-t)\psi_2) \leq tJ(\psi_1) + (1-t)J(\psi_2).$$

That is, if the transform $\pi(\psi)$ is convex for ψ then the total variation is convex for ψ. It remains to show that the function

$$\pi(\psi) = \tau_{\boldsymbol{a}}(\psi) + \theta||\psi - \zeta||^2_{L^2(\Omega)}, \tag{5.26}$$

where ζ is as described above, is convex. Proceeding as in the previous case, we calculate the second derivative

$$\frac{\partial^2 \pi}{\partial \psi^2} = 2\theta - 2j(\psi), \qquad j(\psi) := \frac{2(a_1 + 2a_4)a_2 E(1 - a_2 E)}{(1 + a_2 E)^3 a_3^2}$$

We would like to find the upper bound of this function. We consider the limits

$$\lim_{\psi \to -\infty} j(\psi) = \lim_{\psi \to -\infty} \left(\frac{2(a_1 + 2a_4)a_2 E(1 - a_2 E)}{(1 + a_2 E)^3 a_3^2} \right) = 0,$$

$$\lim_{\psi \to \infty} j(\psi) = \lim_{\psi \to \infty} \left(\frac{2(a_1 + 2a_4)a_2 E(1 - a_2 E)}{(1 + a_2 E)^3 a_3^2} \right) = 0,$$

which are equal to zero. We now find the extrema

$$\frac{\partial j}{\partial \psi} \;=\; -8(a_1 + 2a_4)a_2 E \frac{a_2^2 E^2 - 4a_2 E + 1}{a_3^3 (1 + a_2 E)^4} \;=\; 0 \;\Leftrightarrow\; \psi \;=\; \frac{-a_3}{2} \frac{2 \pm \sqrt{3}}{a_2}$$

which takes values of j given by

$$j\left(\frac{-a_3}{2} \frac{2 \pm \sqrt{3}}{a_2} \right) = -\frac{2(a_1 + 2a_4)(2 \pm \sqrt{3})\left(1 \pm \sqrt{3}\right)}{\left(3 \pm \sqrt{3}\right)^3 a_3^2}$$

It is easy to observe that a positive value is obtained at $\psi = -a_3(2 - \sqrt{3})/2a_2$ and a negative value is obtained at $\psi = -a_3(2 + \sqrt{3})/2a_2$. We can therefore conclude that the values of j lie in the range

$$\left[-\frac{2(a_1 + 2a_4)(3\sqrt{3} + 5)}{\left(3 + \sqrt{3}\right)^3 a_3^2}, \frac{2(a_1 + 2a_4)(3\sqrt{3} - 5)}{\left(3 - \sqrt{3}\right)^3 a_3^2} \right],$$

so that $\partial^2 \pi(\psi)/\partial \psi^2 = 2\theta - 2j(\psi) \geq 0$ as required. ∎

5.5 Experimental Results

Our experimental tests are aimed at showing the effectiveness in restoration of our Algorithm 14 in comparison with Vogel's positivity method [13, 193], the projection method [44] and other methods without positivity constraints. We also compare with unconstrained (and partly constrained) models which have the constraint applied at the end by truncation or scaling.

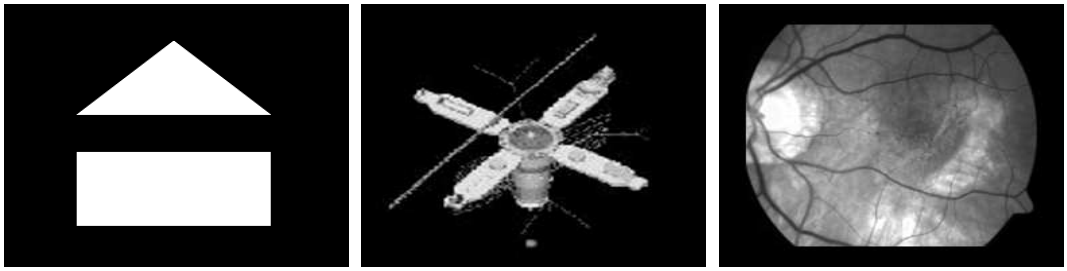
5.5.1 Methods and Test Images

Specifically, in tables and figures, we denote the compared methods by these abbreviations:

- ROF: the well-known model (5.2) without positivity constraint.
- ROF$_{Thr}$: the well-known model (5.2) with positivity and upper limit constraints applied at the end by truncation.
- ROF$_{Sca}$: the well-known model (5.2) with positivity and upper limit constraints applied at the end by scaling.
- Vogel: the non-negatively constrained restoration model by [13].
- Vogel$_{Thr}$: the non-negatively constrained restoration model by [13] with upper limit constraint applied at the end by truncation.
- Vogel$_{Sca}$: the non-negatively constrained restoration model by [13] with upper limit constraint applied at the end by scaling.
- Proj: the constrained projection model by [44].
- New$_5^1$: Algorithm 14 for model (5.5).
- New$_5^2$: Algorithm 14 to solve the minimisation of (5.9).
- New$_5^3$: Algorithm 15 for model (5.14).
- New$_5^4$: Algorithm 14 to solve (5.13) followed by Algorithm 14 to solve (5.7) using the solution of (5.13) as the initial estimate.
- New$_5^5$: Algorithm 14 to solve (5.7) using the solution given by Vogel as the initial estimate.
- New$_5^6$: Algorithm 16 for model (5.19).
- New$_5^7$: Algorithm 16 for model (5.19) followed by projection of the solution to the range $[0, a_1]$. It can be noticed in the results, that this final projection makes very little difference to the result.

We use "Received" to mean the received image z.

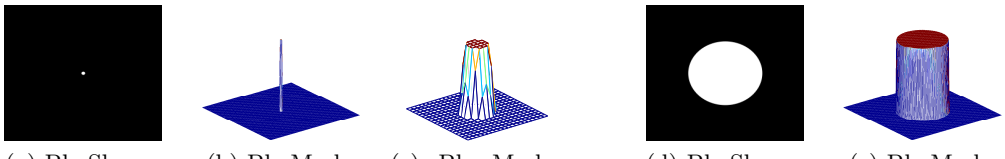
Seven sets of experimental results using 3 test images: the box-triangle image (Im_1), the satellite image (Im_2) and the retina image (Im_3) are selected; see Figure 5.2. For the transform $u = T(\psi)$, we choose $a_1 = 1, 1.08, 255$ and $a_4 = 10^{-2}, 10^{-2}, 0.5$ respectively for the 3 test images. For the blurring model (5.1), we have considered small and large



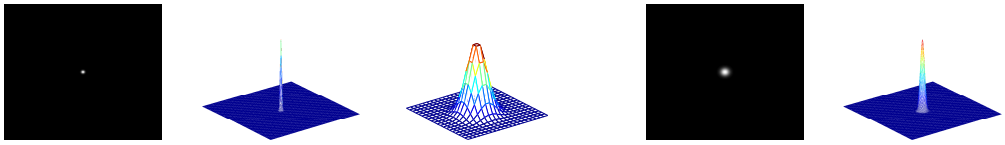
Im_1 - Box-Triangle Image Im_2 - Satellite Image Im_3 - Retina Image

Figure 5.2: Test case images.

levels of motion blur (Bl_1 and Bl_2 respectively) and small and large levels of Gaussian blur (Bl_3 and Bl_4 respectively); see Figure 5.3.



(a) Bl_1 Shape (b) Bl_1 Mesh (c) Bl_1 Mesh Close-up (d) Bl_2 Shape (e) Bl_2 Mesh



(f) Bl_3 Shape (g) Bl_3 Mesh (h) Bl_3 Mesh Close-up (i) Bl_4 Shape (j) Bl_4 Mesh

Figure 5.3: PSFs used for test cases. Images (a)-(c) show Bl_1 - small motion blur, images (d)-(e) show Bl_2 - large motion blur, images (f)-(h) show Bl_3 - small Gaussian blur, and images (i)-(j) show Bl_4 - large Gaussian blur.

5.5.2 Error Measures

There are several common measures for testing the quality of the restored image, some of which have already been introduced in §4.1.3. Here, we recall those error measures and introduce some additional error measures used in this chapter. We let u_{true} denote the true image, u the restored image, z the received image and let m and n be the number of pixels horizontally and vertically respectively. Then we have:

- Mean Squared Error (MSE) is given by

$$MSE = \frac{1}{mn} \sum_{x,y} (u_{\text{true}}(x,y) - u(x,y))^2$$

  and Root Mean Squared Error (RMSE) is given by

$$RMSE = \sqrt{MSE}.$$

- Signal-to-Noise Ratio (SNR) in dB, introduced in §4.1.3, is given by

$$SNR = 10 \log_{10} \left(\frac{\sum_{x,y} |u_{\text{true}}(x,y)|^2}{\sum_{x,y} |u_{\text{true}}(x,y) - u(x,y)|^2} \right).$$

- Peak Signal-to-Noise Ratio (PSNR), introduced in §4.1.3, is given by

$$PSNR = 20 \log_{10} \left(\frac{\max_{x,y} |u_{\text{true}}(x,y)|}{RMSE} \right).$$

- Improved Signal-to-Noise Ratio (ISNR) is given by the difference in PSNR of the restored image u and the received image z:

$$ISNR = PSNR(u) - PSNR(z).$$

Note that the RMSE is given by the L^2-norm of the difference between the true image and the restored image divided by the total number of pixels, i.e. $RMSE = (1/mn)||u_{\text{true}} - u||_{L^2(\Omega)}$ as seen above. Given astronomical images and images with significant amounts of black space, it is typically more common to use the L^1-norm. We expect that these may provide more accurately descriptive measures of our data and the impact of the model in terms of non-negativity. We therefore propose the measures

- L^1 Error given by

$$Er_1 = ||u_{\text{true}} - u||_{L^1(\Omega)} = \frac{1}{mn} \sum_{x,y} |u_{\text{true}}(x,y) - u(x,y)|.$$

- A version of PSNR using the L^1-norm of the difference between the true image and the restored image is given by

$$Er_2 = 20 \log_{10} \left(\frac{\max_{x,y} |u_{\text{true}}(x,y)|}{Er_1} \right).$$

5.5.3 Result Sets

Model (5.1) with Gaussian blur. Result set 1 uses Im_1 corrupted by Gaussian blur to demonstrate the effectiveness of the model in keeping the intensity values of the image constrained. We see in Figure 5.4 and Table 5.1 that [13] keeps the image positive

but allows some points to take intensity values which are outside of the expected range, while [44] and the new models successfully keep the intensity values positive and within the expected range at all points.

Model (5.1) with Motion blur. Result set 2 consists of Im_2 and Im_3 corrupted by small motion or small Gaussian blur. We see in Figure 5.5 and Tables 5.2–5.3 that for images corrupted by small levels of blur the results are competitive between the models. Error values are improved but visual quality is similar.

Model (5.1) with Heavy blurs. Result set 3 consists of Im_2 corrupted by larger levels of blur (Bl_2 and Bl_4). We see in Figure 5.6 and Table 5.4 that that results are improved visually and in the error values for the new model in the case of Bl_2. For Bl_4, the Transform Model appears to be a closer approximation but the error values are similar.

Model (5.1) with Blur and a varying level of noise. Result set 4 consists of Im_2 corrupted by Bl_3 and varying amounts of noise (1% and 50%). We see in Figure 5.7 and Table 5.5 that visually the Transform model offers some improvement in quality while the error values are similar.

Model (5.1) by Algorithm 14 with alternative linerisation (5.8). Result set 5 shows in Figure 5.8 and Table 5.6 the results using the linearised Transform model. We can see that for the same quality of the restored image, the cpu time is improved.

Algorithm 14 combined with Vogel's model. Result set 6 shows in Figure 5.9 and Table 5.7 examples using the received image as the initial estimate and the results of Vogel's model as the initial estimate. We can see that this technique is useful for restoring the PSF given the image. In the case of the motion blur example, the cpu time is significantly improved and in the case of Gaussian blur, the error value is improved. In all cases, the visual quality is adequate.

Model (5.2) with Blurs. Now we consider the solution of model (5.3) for k. Result set 7 consists of motion and Gaussian blur PSFs which are regarded as being blurred by Im_2. The task here is to recover the PSF given the true image. As the initial estimate, rather than taking the received data z as the initial estimate (since it is not expected to be a good approximation of the true kernel) we make an estimate of the kernel based on observation of the received data. We see in Figure 5.10 that in both cases, each of the models are able to obtain good approximations of the kernel, however ROF is unable to retain non-negativity in both cases and Vogel, while successfully ensuring positivity of the approximated kernel, struggles to get correct smaller values as well as larger values whereas the transform model is able keep the values close to zero as well as ensuring positivity of the result.

Finally, to simultaneously restore both u and k in the so-called blind deconvolution problem, the TV based model by [54] is the following

$$\min_u \int_\Omega (u * k - z)^2 \,\mathrm{d}\Omega + \alpha_1 \|u\|_{TV}^\beta + \alpha_2 \|k\|_{TV}^\beta, \quad \text{s. t. } k \geq 0, \int_\Omega k(s,t)\,\mathrm{d}s\,\mathrm{d}t = 1, \quad (5.27)$$

where $\alpha_1, \alpha_2 > 0$. Related studies can be found in [2, 33, 198, 179, 206]. In other experiments, we have tried double transforms which appear to improve the robustness. This model is considered further in Chapter 6.

Method	cpu	rmse	Er_1	Er_2	snr	psnr	isnr
Received	n/a	0.0771	39.35	16.27	15.79	22.26	n/a
ROF	32.2	0.0589	27.39	19.41	18.33	24.60	2.33
ROF_{Thr}	32.2	0.0528	26.40	19.73	19.15	25.54	3.28
ROF_{Sca}	32.2	0.1573	55.28	13.31	8.35	16.07	-6.20
Vogel	37.8	0.0320	17.24	23.43	23.61	29.89	7.63
Vogel_{Thr}	37.8	0.0303	17.24	23.43	24.02	30.36	8.10
Vogel_{Sca}	37.8	0.0765	25.04	20.19	14.75	22.33	0.07
Proj	33.5	0.0378	18.78	22.69	22.12	28.46	6.20
New_5^1	59.7	**0.0149**	6.86	31.43	**30.28**	**36.55**	**14.29**
New_5^3	26.3	0.0236	11.31	27.10	26.25	32.53	10.26
New_5^6	15.76	0.0241	**4.45**	**35.20**	26.02	32.35	10.09
New_5^7	15.76	0.0241	**4.45**	**35.20**	26.02	32.35	10.09

Table 5.1: Result Set 1 - Error values for Im_1 corrupted by Gaussian blur with no Noise. We can see that the error values are improved when using the Transform models and cpu time is improved by using New_5^3–New_5^7. As designed, the results of New_5^3–New_5^7 are very similar, showing that the additional term does not have a considerable effect on results. It is also evident from New_5^6–New_5^7 that the final projection does not have a significant impact on the results.

5.6 Conclusion

We have presented models to reconstruct images and PSFs and demonstrated that they can ensure positivity through introducing a transform and also keep the intensities of the restored data within the appropriate range. We have also demonstrated that the model offers competitive results in the case of small levels of blur and noise but much improved results in the case of corruption by larger levels of blur and noise. This model is particularly effective in giving a close approximation of the kernel (in the case where the image is known) which is of great importance in the case of blind deblurring. The transform idea is applicable potential to a class of other variational models. Since non-negativity is a significant criterion for blind deblurring models, we consider such applications in Chapter 6.

5.A Selection of Parameters in $T(\psi)$

The parameter a_1 is easily chosen, assuming knowledge of the bits-per-sample (bps) value of the true image and the blurred image. This will typically be between 1 and 255 for images of bps 1 to 8 respectively, but can be quite low for the kernel. For example, a fairly compact-radius out-of-focus blur may have a kernel value upper limit

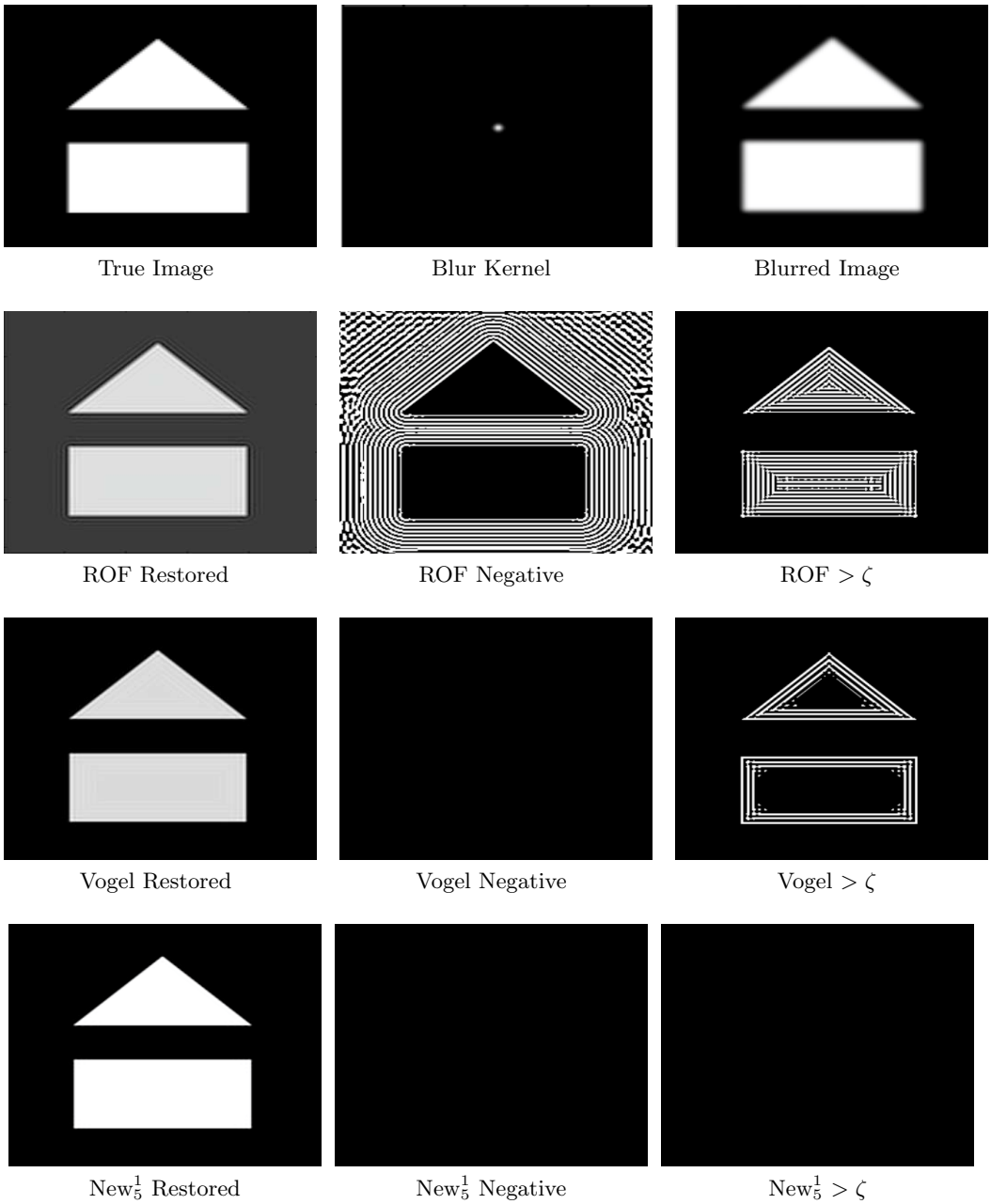


Figure 5.4: Result Set 1: Restoring Im_1 corrupted by Bl_3 with no noise. From top to bottom, we have: 1) the true image, kernel, and corrupted data; 2) the result using the ROF method; 3) the result using Vogel's method; 4) the result using the Transform method. From left to right, we have (on rows 2-4): 1) the restored image; 2) the negative values of the restored image in white; 3) the points where the intensity values are greater than the expected upper limit in white. Note that the Transform method and Vogel's method can both ensure positivity but the transform method can control the upper bound of the intensity range.

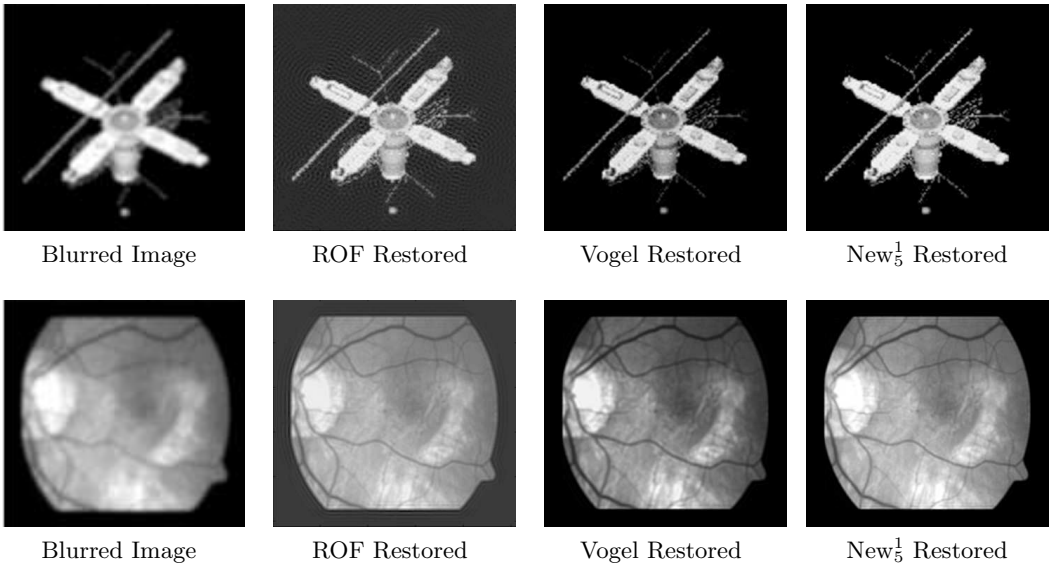


Figure 5.5: Result Set 2 - restoring images Im_2 and Im_3 corrupted by small motion blur Bl_1 or small Gaussian blur Bl_3. In some cases the results from the Transform model appear sharper than other models and more small detail is visible.

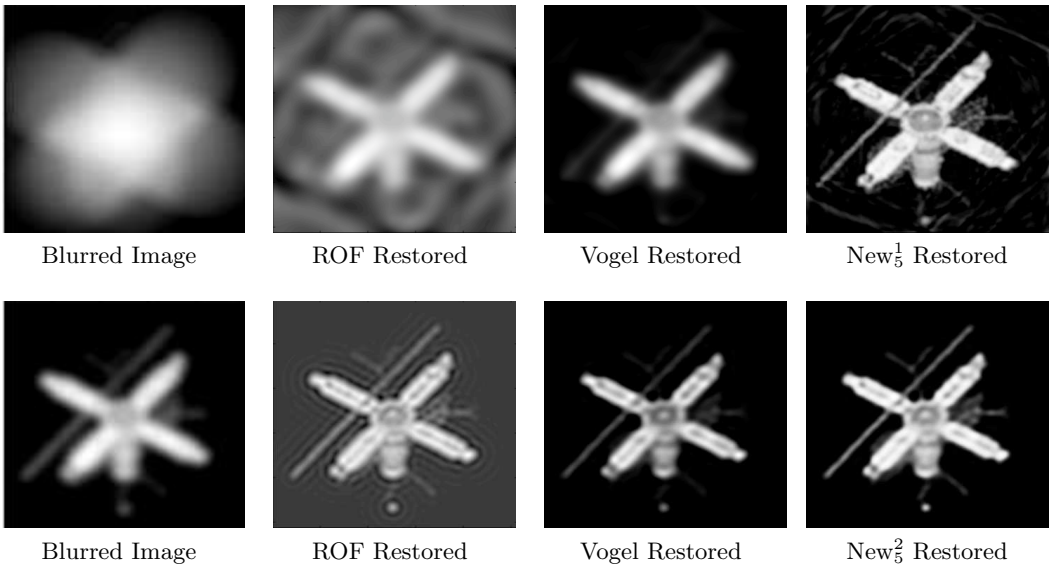


Figure 5.6: Result Set 3 - Restoring Im_2 corrupted by Bl_2 (top line) and by Bl_4 (bottom line). We can see a significant improvement in the result from the Transform method in the case of corruption by Bl_2, and the results are competitive in the case of Bl_4.

Error values for Im_2 corrupted by Bl_1							
Method	cpu time	rmse	Er_1	Er_2	snr	psnr	isnr
Received	n/a	0.0478	11.18	27.19	14.47	26.41	n/a
ROF	40.6	0.0211	6.31	32.17	21.93	33.51	7.10
ROF_{Thr}	40.6	0.0183	5.05	34.09	23.15	34.74	8.33
ROF_{Sca}	40.6	0.0932	25.81	19.93	9.76	20.62	-5.79
Vogel	31.9	0.0107	3.08	38.39	27.79	39.39	12.98
$Vogel_{Thr}$	31.9	0.0107	3.08	38.39	27.79	39.39	12.98
$Vogel_{Sca}$	31.9	0.0478	3.12	38.27	27.85	39.38	12.98
Proj	16.2	0.0054	1.37	45.40	33.76	45.33	18.92
New_5^1	38.7	0.0036	**0.96**	**48.54**	37.31	48.88	22.47
New_5^3	12.8	0.0051	1.36	45.51	34.24	45.81	19.40
New_5^6	12.6	**0.0033**	1.16	46.89	**38.27**	**49.75**	**23.34**
New_5^7	12.6	**0.0033**	1.16	46.89	**38.27**	**49.75**	**23.34**
Error values for Im_3 corrupted by Bl_1							
Method	cpu time	rmse	Er_1	Er_2	snr	psnr	isnr
Received	n/a	0.0362	19.19	22.50	21.39	28.82	n/a
ROF	34.7	0.0178	9.09	29.00	27.62	34.97	6.16
ROF_{Thr}	34.7	0.0164	8.52	29.55	28.33	35.69	6.88
ROF_{Sca}	34.7	0.0557	27.47	19.39	17.70	25.08	-3.74
Vogel	25.2	0.0084	3.95	36.24	34.14	41.51	12.70
$Vogel_{Thr}$	25.2	0.0084	3.95	36.24	34.14	41.52	12.70
$Vogel_{Sca}$	25.2	0.0113	4.54	35.02	31.39	38.90	10.09
Proj	17.5	0.0056	1.86	42.79	37.62	44.97	16.16
New_5^1	66.0	**0.0020**	0.94	48.67	**46.57**	**54.03**	**25.21**
New_5^3	12.9	0.0044	1.30	45.86	39.69	47.04	18.23
New_5^6	14.96	0.0027	**0.80**	**50.07**	44.03	51.34	22.52
New_5^7	14.96	0.0027	**0.80**	**50.07**	44.03	51.34	22.52

Table 5.2: Result Set 2 - Error values for images Im_2 and Im_3 corrupted by Bl_1. It can be noticed that error values are improved using the Transform models. While cpu time is higher than that of competing models, New_5^3–New_5^7 can reduce cpu time while retaining similar or improved PSNR. As designed, the results of New_5^3–New_5^7 are very similar, showing that the additional term does not have a considerable effect on results. It is also evident from New_5^6–New_5^7 that the final projection does not have a significant impact on the results.

Error values for Im_2 corrupted by Bl_3							
Method	cpu time	rmse	Er_1	Er_2	snr	psnr	isnr
Received	n/a	0.0562	12.72	26.08	12.96	25.01	n/a
ROF	36.4	0.0263	7.01	31.25	19.99	31.59	6.59
ROF_{Thr}	36.4	0.0249	6.51	31.89	20.46	32.07	7.06
ROF_{Sca}	36.4	0.1221	33.49	17.67	7.52	18.27	-6.74
Vogel	32.2	0.0233	6.38	32.07	21.01	32.65	7.64
$Vogel_{Thr}$	32.2	0.0233	6.38	32.07	21.01	32.65	7.64
$Vogel_{Sca}$	32.2	0.0236	6.31	32.16	21.07	32.55	7.54
Proj	17.4	0.0203	5.52	33.33	22.27	33.87	8.86
New_5^1	45.8	**0.0142**	**4.18**	**35.73**	**25.39**	**36.97**	**11.96**
New_5^3	13.0	0.0172	5.05	34.10	23.70	35.28	10.27
New_5^6	16.2	0.0156	4.70	34.73	24.58	36.13	11.12
New_5^7	16.2	0.0156	4.70	34.73	24.58	36.13	11.12
Error values for Im_3 corrupted by Bl_3							
Method	cpu time	rmse	Er_1	Er_2	snr	psnr	isnr
Received	n/a	0.0422	24.01	20.56	20.04	27.49	n/a
ROF	35.2	0.0236	14.79	24.76	25.17	32.54	5.05
ROF_{Thr}	35.2	0.0226	14.35	25.03	25.56	32.93	5.44
ROF_{Sca}	35.2	0.0855	42.65	15.57	14.18	21.36	-6.12
Vogel	23.8	0.0169	9.77	28.37	28.06	35.44	7.95
$Vogel_{Thr}$	23.8	0.0169	9.77	28.37	28.08	35.45	7.96
$Vogel_{Sca}$	23.8	0.0240	10.98	27.36	24.68	32.39	4.90
Proj	25.1	0.0177	11.44	26.99	27.68	35.05	7.56
New_5^1	87.4	**0.0127**	**7.84**	**30.27**	**30.53**	**37.89**	**10.40**
New_5^3	13.1	0.0171	11.04	27.31	27.99	35.35	7.87
New_5^6	16.3	0.0147	9.10	28.99	29.35	36.66	9.17
New_5^7	16.3	0.0147	9.10	28.99	29.35	36.66	9.17

Table 5.3: Result Set 2 - Error values for images Im_2 and Im_3 corrupted by Bl_3. It can be noticed that error values are improved using the Transform model. While cpu time is higher than that of competing models, New_5^3–New_5^7 can reduce cpu time without a significant reduction in PSNR. As designed, the results of New_5^3–New_5^7 are very similar, showing that the additional term does not have a considerable effect on results. It is also evident from New_5^6–New_5^7 that the final projection does not have a significant impact on the results.

Error values for Im_2 corrupted by Bl_2							
Method	cpu time	rmse	Er_1	Er_2	snr	psnr	isnr
Received	n/a	0.22	63.17	12.80	-2.13	13.66	n/a
ROF	2.01	0.13	33.77	18.24	6.08	18.31	4.65
Vogel	16.02	0.11	26.55	20.33	7.30	19.71	6.05
New_5^1	47.64	**0.06**	**14.87**	**25.36**	**14.03**	**25.41**	**11.75**
New_5^2	30.18	0.07	18.68	23.38	11.77	23.33	9.67
New_5^4	13.65	0.10	24.32	21.09	9.03	20.87	7.21
Error values for Im_2 corrupted by Bl_4							
Method	cpu time	rmse	Er_1	Er_2	snr	psnr	isnr
Received	n/a	0.0909	21.40	21.56	8.04	20.83	n/a
ROF	54.8	0.0596	14.98	24.66	12.72	24.49	3.66
Vogel	37.9	0.0565	13.00	25.88	13.11	24.96	4.13
New_5^1	31.3	**0.0489**	**11.72**	**26.78**	**14.45**	**26.22**	**5.39**

Table 5.4: Result Set 3 - Error values for Im_2 corrupted by Bl_2 and Bl_4. There is a noticeable improvement in the case of and while the results for Bl_4 are competitive, the transform is slightly improved over competing models.

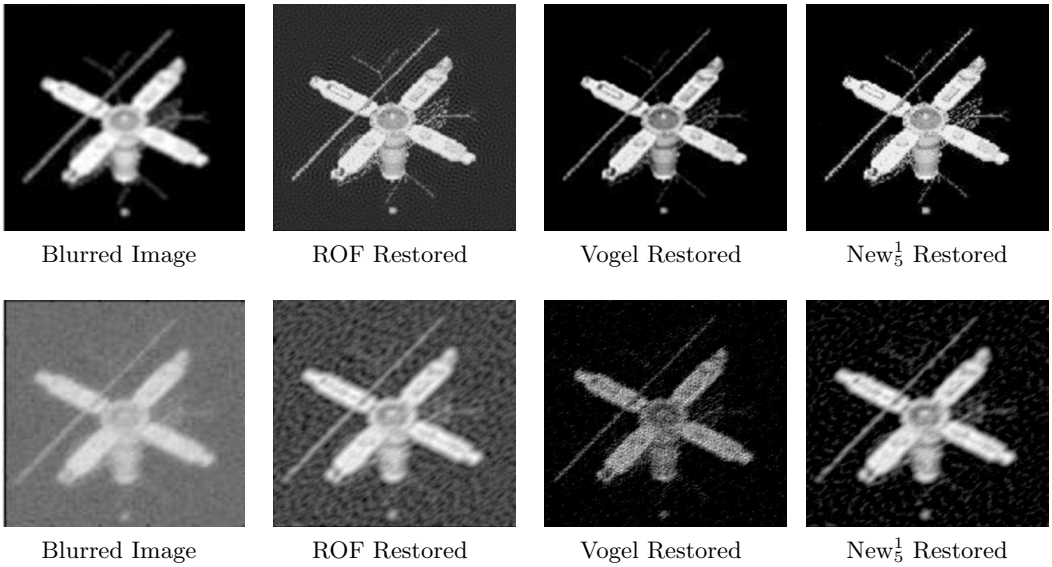
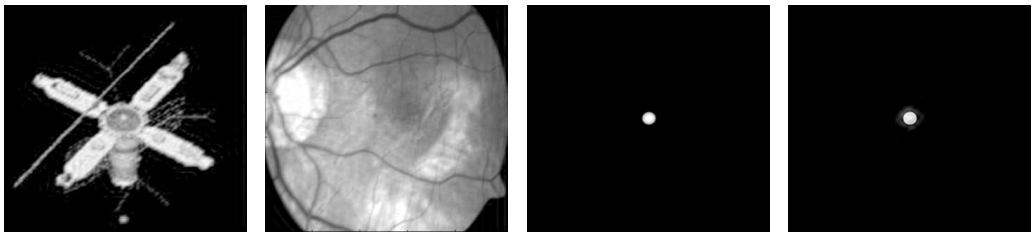


Figure 5.7: Result Set 4 - Restoring Im_2 corrupted by Bl_3 and 1% noise (top row) and 50% noise (bottom row). We can see that visually the Transform method appears to give improved results for weaker and stronger levels of noise.

Error values for Im_2 corrupted by Bl_3 and 1% noise.							
Method	cpu time	rmse	Er_1	Er_2	snr	psnr	isnr
Received	n/a	0.0479	11.34	27.07	14.49	26.40	n/a
ROF	42.1	0.0304	7.85	30.26	18.82	30.35	3.95
Vogel	12.1	0.0237	6.41	32.03	20.80	32.52	6.12
New_5^3	4.9	**0.0196**	**5.73**	**33.01**	**22.61**	**34.15**	**4.85**
Error values for Im_2 corrupted by Bl_3 and 50% noise.							
Method	cpu time	rmse	Er_1	Er_2	snr	psnr	isnr
Received	n/a	0.0639	60.71	13.14	-13.24	12.59	n/a
ROF	15.00	0.0783	19.58	22.97	11.50	22.77	10.19
Vogel	5.64	0.0980	22.91	21.61	9.72	20.86	8.27
New_5^1	55.76	**0.0718**	**17.12**	**24.14**	**12.20**	**23.52**	**10.94**

Table 5.5: Result Set 4 - Error values for Im_2 corrupted by Bl_2 and varying amounts of noise. We can see that the Transform model can offer improved results, particularly for larger levels of noise.

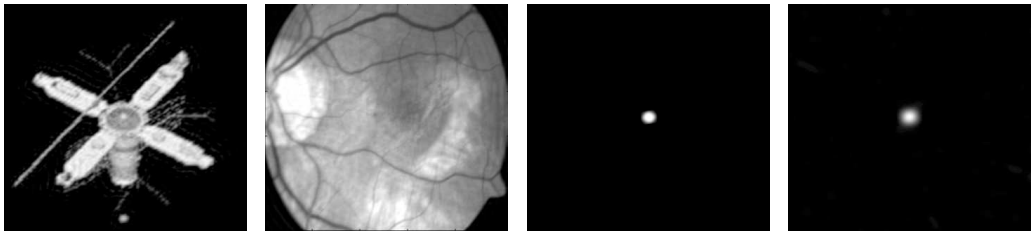


Im_2 Im_3 Bl_1 Bl_3

Figure 5.8: Result Set 5: Restored images and PSFs using the Linearised Transform method. The received data from which Im_2 and Im_3 were restored was corrupted by Bl_1, and the received data from which Bl_1 and Bl_3 were restored was corrupted by Im_2. We can see that the linearisation does not affect the visual quality significantly.

	Transform Model (New_5^1)		Linearised New_5^1	
Image	psnr	cpu time	psnr	cpu time
Im_2	30.32	60.07	30.54	**34.10**
Im_3	35.62	83.05	35.51	**35.09**
Bl_1	38.63	72.25	38.24	**51.00**
Bl_3	39.56	82.77	37.59	**47.84**

Table 5.6: Result Set 5: Error values and cpu time for restoring images Im_2 and Im_3 as well as PSFs Bl_1 and Bl_3 using the Transform method and the Linearised Transform method. We can see that the quality of the restored image is not significantly different for each case but the cpu time is improved using the Linearised Transform method.



Im_2 Im_3 Bl_1 Bl_3

Figure 5.9: Result Set 6: Restored images and PSFs using the Linearised Transform method with the result of Vogel's method as the initial estimate.

	New_5^1		New_5^5	
Image	psnr	cpu time	psnr	cpu time
Im_2	30.54	**34.10**	**30.61**	39.79
Im_3	35.51	**35.09**	**35.71**	46.83
Bl_1	38.24	51.00	**38.80**	**27.59**
Bl_3	37.59	**47.84**	**42.53**	51.14

Table 5.7: Result Set 6: Error values and cpu times for restoring images Im_2 and Im_3 and PSFs Bl_1 and Bl_3 using the Linearised Transform method with the received data z as the initial estimate (New_5^1) and the result from Vogel's method as the initial estimate (New_5^5). The cpu time is rarely lower when using the closer initial estimate but the image quality is improved in all cases.

of 10^{-2}. While a larger value of a_1 should still give a good approximation, it is essential that a_1 be at least as large as the maximum image intensity value or kernel value and advisable that it be close to this. The parameter a_4 should be chosen in proportion to a_1. Typically, $a_4 = a_1/255$ is a sensible value.

We attempt to select the remaining parameters a_2 and a_3 in order to control the upper and lower bounds of ψ as well as the value of ψ when u is equal to zero. In order to control the bounds, we define a length $\Sigma = \sigma_4 - \sigma_3$ where σ_3 and σ_4 represent two intensity values of ψ. We would then like for $\tau_4 - \tau_3 = T(\sigma_4) - T(\sigma_3) = \Sigma$. From $\psi(\tau) = T^{-1}(\tau) = -\frac{a_3}{2} \ln\left(\frac{a_1 - \tau + a_4}{a_2(\tau + a_4)}\right)$, we have

$$\Sigma = \sigma_4 - \sigma_3 = \psi(\tau_4) - \psi(\tau_3) \tag{5.28}$$

$$= \frac{a_3}{2} \ln\left(\frac{(a_1 - \tau_3 + a_4)(\tau_4 + a_4)}{(\tau_3 + a_4)(a_1 - \tau_4 + a_4)}\right). \tag{5.29}$$

So, assuming we fix Σ, τ_3, τ_4, a_1 and a_4, we have

$$a_3 = \frac{2\Sigma}{\ln\left(\frac{(a_1 - \tau_3 + a_4)(\tau_4 + a_4)}{(\tau_3 + a_4)(a_1 - \tau_4 + a_4)}\right)}$$

For our model, we fix the width $\Sigma = \tau_4 - \tau_3$ (see Figure 5.11) and let $\tau_4 = a_1 - \tau_3$.

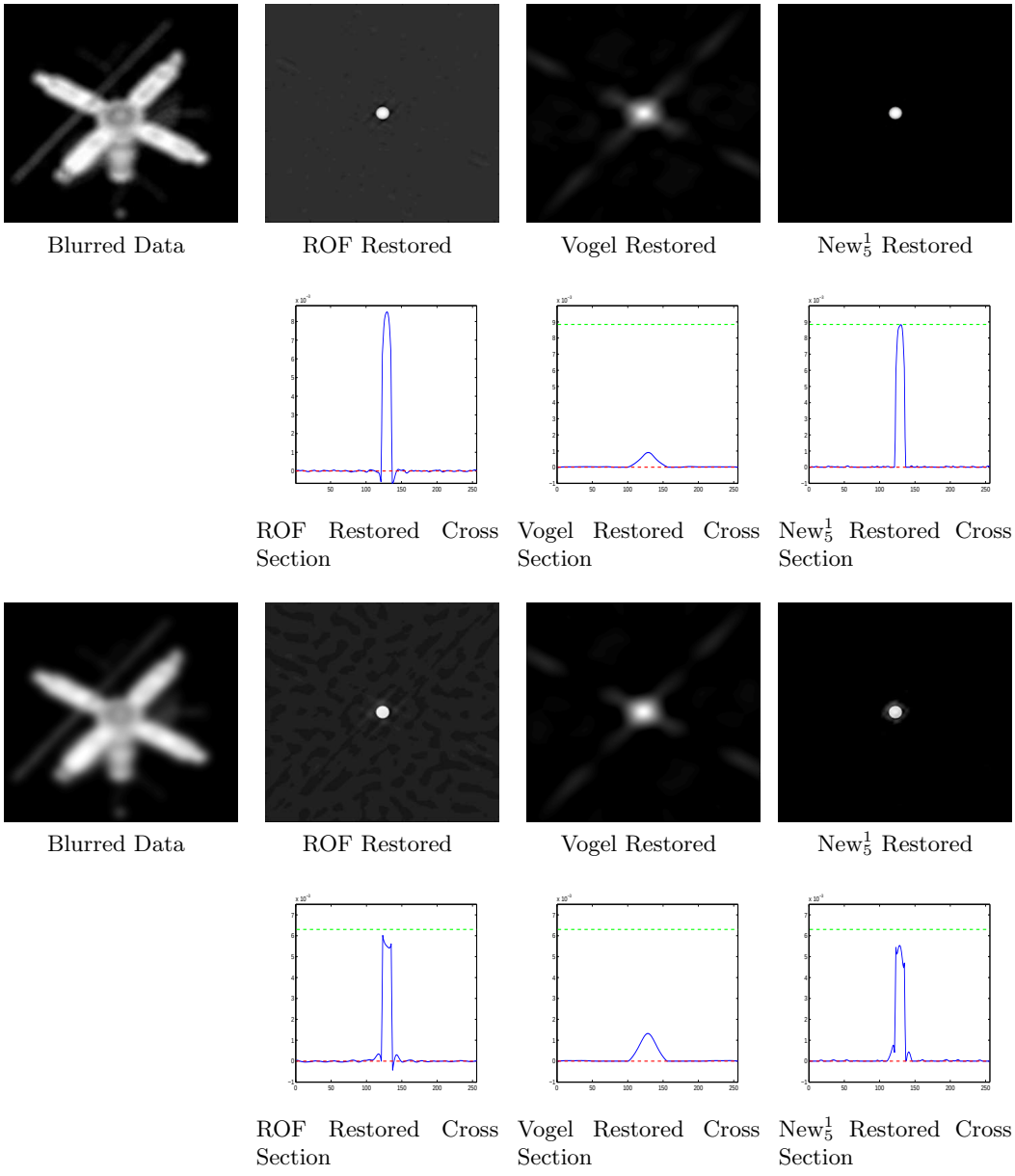
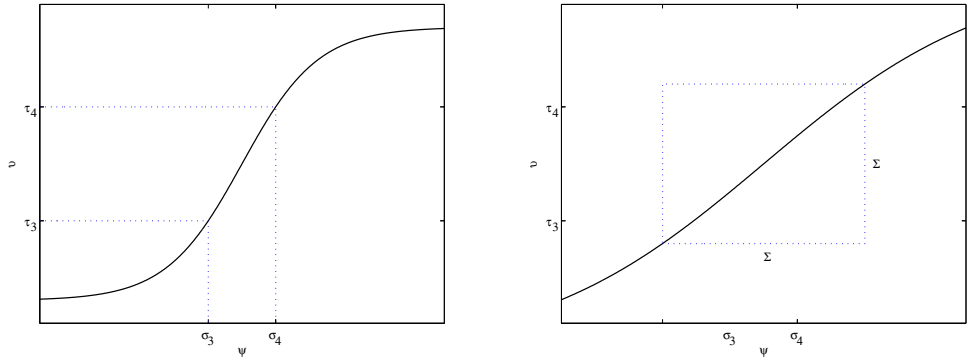


Figure 5.10: Result Set 7 - Restoring Bl_1 (1st and 2nd rows) and Bl_2 (3rd and 4th rows) corrupted by Im_1 restored using TV restoration (ROF), Vogel's model (Vogel) and the transform model (New$_5^1$). In the cross-section images, the blue line is the restored image, the red dashed line is the lower bound of the true blur function and the green dashed line is the upper bound of the true blur function. Of the three approximations, as demonstrated in the cross-section images on the 2nd and 4th rows, the TV model gives many negative values in the approximation both kernels, and Vogel's model has no negative values but struggles to get a close approximation while the transform model does a good job.

Then, from 5.29, we have

$$a_3 = \frac{2(\tau_4 - \tau_3)}{\ln\left(\frac{(a_1-\tau_3+a_4)(\tau_4+a_4)}{(\tau_3+a_4)(a_1-\tau_4+a_4)}\right)} = \frac{a_1 - 2\tau_3}{\ln\left(\frac{a_1-\tau_3+a_4}{\tau_3+a_4}\right)}.$$

The only remaining parameter which a_3 is dependent on and which has not already been decided is τ_3. We find that $\tau_3 = a_1/4$ is adequate for the transform.



(a) (b)

Figure 5.11: Graph of Transform $u = T(\psi)$. On the left, (a) demonstrates the correspondence between the σ and τ parameters and on the right, (b) shows that the differences $\sigma_4 - \sigma_3$ and $\tau_4 - \tau_3$ are equal to Σ.

We may use the parameter a_2 to control the value of ψ at $u = T(\psi) = 0$. We consider two cases: the first given by $T(\psi) = a_1/2$ and the second given by $T(\psi) = \tau_1$ at $\psi = 0$ where τ_1 is the lower bound of ψ. The first option will make the graph pass through zero at the midpoint of the intensity values and the second will make all values of ψ naturally positive since the lower bound of ψ will be equal to zero. Letting $u = T(\psi)$

$$u = \frac{a_1 + 2a_4}{1 + a_2 e^{\frac{-2\psi}{a_3}}} - a_4.$$

Rearranging, we have

$$a_2 = \frac{a_1 + a_4 - u}{e^{\frac{-2\psi}{a_3}}(u + a_4)}$$

and so for the first case, we have

$$a_2 = \frac{a_1 + a_4 - a_1/2}{a_1/2 + a_4} = \frac{a_1/2 + a_4}{a_1/2 + a_4} = 1,$$

and for the second case, we have

$$a_2 = \frac{a_1 + a_4 - \tau_1}{\tau_1 + a_4}.$$

In application, either of these will be sufficient to recover the image with similar

results. In the case of the kernel, better results are obtained with $a_2 = 1$. It is there advised therefore that $a_2 = 1$ is the appropriate value for this parameter.

In summary, once a_1 and a_4 are defined, the other quantities in the transform $T(\psi) = \frac{a_1+2a_4}{1+a_2 e^{\frac{-2\psi}{a_3}}} - a_4$ can be determined automatically assuming that $\tau_3 = a_1/4$ and $a_2 = 1$ are acceptable.

5.B Derivation of Total Variation Regularisation of a Transformed Function

Here, we present the derivation total variation regularisation term of a transformed function for the associated Euler-Lagrange equation. Letting

$$E_1 = E_1(\psi) = a_2 e^{\frac{-2\psi}{a_3}},$$

we have

$$\begin{aligned}\min_{\psi} TV(T_{\boldsymbol{a}}(\psi)) &= \int_\Omega \frac{2(a_1+2a_4)E_1\sqrt{\psi_x^2+\psi_y^2}}{(1+E_1)^3 a_3^2(\psi_x^2+\psi_y^2)}\left(\psi_x^2\phi(-2+2E_1)\right.\\ &+ \psi_x\phi_x(a_3+a_3E_1) + \psi_y^2\phi(-2+2E_1)\\ &+ \left.\psi_y\phi_y(a_3+a_3E_1)\right).\end{aligned}$$

Now noting that $\nabla\psi\cdot\nabla\psi = \psi_x^2+\psi_y^2$ and letting

$$E_2 = E_2(\psi) = \frac{2(a_1+2a_4)E_1\sqrt{\psi_x^2+\psi_y^2}}{(1+E_1)^3 a_3^2(\psi_x^2+\psi_y^2)} = \frac{2(a_1+2a_4)E_1\sqrt{\nabla\psi\cdot\nabla\psi}}{(1+E_1)^3 a_3^2(\nabla\psi\cdot\nabla\psi)},$$

we have

$$\begin{aligned}\min_{\psi} TV(T_{\boldsymbol{a}}(\psi) &= \int_\Omega E_2\left(2\phi(E_1-1)(\psi_x^2+\psi_y^2) + a_3(E_1+1)(\psi_x\phi_x+\psi_y\phi_y)\right)\\ &= \int_\Omega E_2\left(2\phi(E_1-1)(\nabla\psi\cdot\nabla\psi) + a_3(E_1+1)(\nabla\psi\cdot\nabla\phi)\right)\\ &= \int_\Omega 2\phi E_2(E_1-1)\nabla\psi\cdot\nabla\psi + \int_\Omega a_3E_2(E_1+1)\nabla\psi\cdot\nabla\phi.\end{aligned}$$

Using Green's theorem $\int_\Omega F\cdot\nabla v = -\int_\Omega \nabla\cdot(Fv) + \int_{\partial\Omega} vF\cdot n$ we have

$$\int_\Omega a_3E_2(E_1+1)\nabla\psi\cdot\nabla\phi = -\int_\Omega \nabla\cdot(a_3E_2(E_1+1)\nabla\psi\phi) + \int_{\partial\Omega}\phi a_3E_2(E_1+1)\nabla\psi\cdot n.$$

Therefore, minimisation of the total variation term leads to

$$2E_2(E_1-1)\nabla\psi\cdot\nabla\psi - \nabla\cdot(a_3E_2(E_1+1)\nabla\psi) = E_2(E_1-1)|\nabla\psi|^2 - \nabla\cdot(a_3E_2(E_1+1)\nabla\psi) = 0$$

with boundary conditions

$$a_3E_2(E_1+1)\nabla\psi\cdot n = 0$$

where we used $|\nabla\psi| = \sqrt{\nabla\psi \cdot \nabla\psi}$ and denoted

$$E_2 = E_2(\psi) = \frac{2(a_1 + 2a_4)E_1|\nabla\psi|}{(1 + E_1)^3 a_3^2 |\nabla\psi|^2}.$$

Thus the above term is simplified to

$$\frac{4E_1(a_1 + 2a_4)(E_1 - 1)|\nabla\psi|}{(1 + E_1)^3 a_3^2} - \nabla \cdot \left(\frac{2(a_1 + 2a_4)E_1}{(1 + E_1)^2 a_3 |\nabla\psi|} \nabla\psi \right) = 0.$$

Chapter 6

A Robust Model for Constrained Blind Image Deblurring

Building on the work of Chapter 5, we now show the second piece of our main work which is a core part of this project and addresses the major challenge of how to impose the extra constraints to ensure uniqueness in the case blind deconvolution. In this chapter, we propose new models and algorithms for implementing blind deconvolution employing the transform based idea for imposing constraints automatically from Chapter 5, avoiding the commonly used thresholding which leads to unsatisfactory solutions. We also present numerical results which demonstrate the robustness of the proposed method for images with either motion or Gaussian blur.

6.1 Introduction

In this chapter, we focus on image deblurring in the blind case where the blur operator is unknown. In this case, we are aiming to reconstruct the true image and the cause of the degradation with no prior information. Of course, if extra information of the blur operator is available, it should be used to derive the so-called semi-blind models [2, 5, 11], considered in more detail in Chapter 7. There exist other approaches [124, 125] beyond the framework that we consider here. Although there exist other approaches [37, 38, 187] for deblurring, we shall focus on the variational framework.

In the paper by Chan and Wong [54], they proposed an energy minimising model, derived partial differential equations by minimising with respect to the image u and the kernel h and presented an alternate minimisation scheme for solving the model. It transpires however that while the partial differential equations are each convex the problem is not jointly convex and so, for example, the negatives of the solutions are also solutions to the problem, since $(-h) * (-u) = (-1)(-1)h * u = h * u$ where $*$ represents the operation of convolution. The solution given was to impose constraints at each outer iteration which ensure that the kernel and the image are non-negative, the point spread function is symmetric and the kernel has a unit integral. Such artificial and at times very intrusive constraints, if imposed separately (not simultaneously) from the

main equations as in [54, 103], can however have a very negative effect on the results. Several ideas have been proposed to address this, including leaving out the constraints altogether and trusting that the model will give a good result. This however is not reliable since it does not give a unique solution.

In this chapter, we aim to remove at least two of the constraints, namely the positivity constraints for the kernel and the image, by imposing the property of non-negativity implicitly in the functional. The end product is a robust image deblurring model; we also present two ways of solving the model. Although it is fair to say that the model by [54] is not yet reliable for general use, there have been more than 500 citations of it and there are several works trying to adapt for and extend to specific applications [38, 184, 165, 198, 185]. Our implicit treatment of positivity (or transformation approach) has enabled the variational formulation to achieve its potential of restoration of both the image and the kernel function.

The rest of this chapter is organised as follows. §6.2 reviews some related variational models. Three test examples are shown to illustrate the Chan-Wong model [54] and highlight the problems and challenges. §6.3 introduces our transformation approach where both the image and the blurring function are reconstructed with non-negativity imposed implicitly. §6.4 considers solution methods of the model resulting from §6.3. In §6.5, we present experimental results, and we present the conclusion of this chapter in §6.6.

6.2 The Inverse Problem and Current Models

In this section, we briefly review some solutions to the blind deblurring problem from §3.4 and some other relevant work. We adopt the problem setting as known in Chapter 5 and the literature [52, 194, 193, 169]. Assume $z = z(x, y)$ is a given image function in domain Ω which may be restored by the additive Gaussian model

$$z = h * u + \eta \tag{6.1}$$

where η is an unknown Gaussian white noise with zero mean, u is the image to be restored and the h is the unknown blurring kernel to be approximated. If we assume that the kernel is spatially invariant, then h becomes a convolution operator. Clearly without regularisation, there is no possibility of finding a unique solution from (6.1) alone.

You and Kaveh [216] proposed a model for simultaneous recovery of both the degradation function and the image, solving the problem

$$\min_{u,h} f_{YK}(u, h) = \frac{1}{2}||h * u - z||^2_{L^2(\Omega)} + \alpha_1 L_1(u) + \alpha_2 L_2(h) \tag{6.2}$$

which includes a fitting term, aiming to keep the restored image and restored kernel close to the true image and kernel respectively, and regularisation for the image and

kernel given by L_1 and L_2. You and Kaveh proposed the H^1 semi-norm for these two terms.

Chan and Wong [54] proposed an improvement to this using the total variation semi-norm for regularisation given by $L_1(u) = \int_\Omega |\nabla u|$ and $L_2(h) = \int_\Omega |\nabla h|$, hence solving

$$\min_{u,h} f_{CW}(u,h) = \frac{1}{2}||h * u - z||^2_{L^2(\Omega)} + \alpha_1 \int_\Omega |\nabla u| + \alpha_2 \int_\Omega |\nabla h| \tag{6.3}$$

where $|\nabla u| = \sqrt{u_x^2 + u_y^2}$. Minimising equation (6.3) with respect to the image u and the kernel h, we obtain the partial differential equations given by

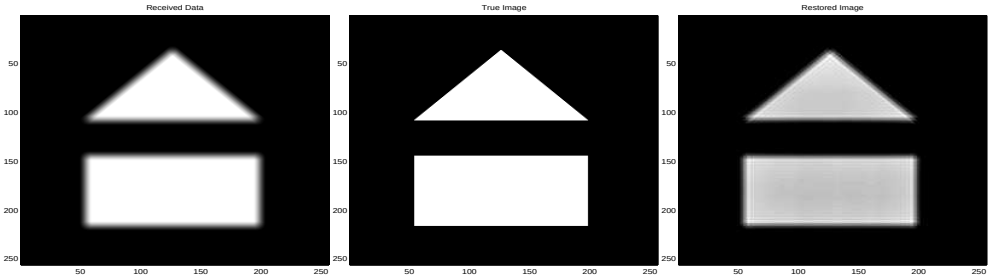
$$\begin{cases} h(-x,-y) * ((h*u)(x,y) - z(x,y)) - \alpha_1 \nabla \cdot \left(\dfrac{\nabla u(x,y)}{|\nabla u(x,y)|_\beta} \right) = 0, \\ u(-x,-y) * ((u*h)(x,y) - z(x,y)) - \alpha_2 \nabla \cdot \left(\dfrac{\nabla h(x,y)}{|\nabla h(x,y)|_\beta} \right) = 0, \end{cases} \tag{6.4}$$

where $|\nabla u(x,y)|_\beta = \sqrt{u_x^2(x,y) + u_y^2(x,y) + \beta^2}$ (similarly for $|\nabla h(x,y)|_\beta$ also) where β is a small positive parameter introduced to avoid division by zero. It is worth noting that as alternatives to the Total Variation semi-norm we may also consider other regularisation terms such as L^2 regularisation, given by $L(u) = \int_\Omega |\nabla u|^2 \, \mathrm{d}\Omega$, the Non-Local TV [217, 213, 219], the Total Generalised Variation (TGV) [116, 22, 21] or the mean curvature [26, 188], as well as others [137, 155, 112, 133].

In order to solve the system, a scheme was proposed which involved alternate minimisation to recover the kernel and the image, including the following constraints which aim to deal with the lack of a unique solution since the system is not jointly convex. This leads to imposing the constraints that the image and kernel should both be positive ($h(x,y) > 0$, $u(x,y) > 0$), the kernel should be symmetric ($h(x,y) = h(-x,-y)$) and the kernel should have a unit integral ($\int_\Omega h(x,y) \, \mathrm{d}\Omega = 1$). These constraints are imposed at each alternate minimisation step as shown in Algorithm 6 on page 68.

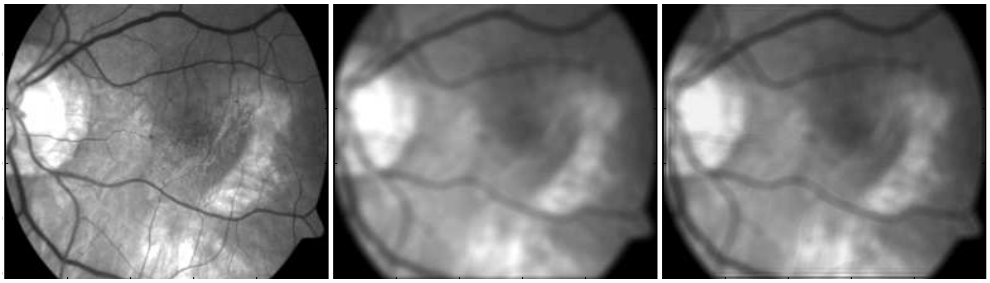
Adding the above 4 constraints ensures a unique solution but introduces inconsistency which is problematic. The algorithm yields a reasonable result for the example given in Figure 6.1c, but the same algorithm gives rise to poor results such as Figures 6.2c and 6.2d due to this inconsistency. We may attempt to improve the results by introducing a small positive parameter κ and implementing an alternative thresholding technique applied to the kernel as described in §4.3. Doing this with a problem dependent parameter κ may offer some improvement (Figure 2e) but it does not always lead to a good solution. Our aim is to satisfy exactly these constraints by achieving the positivity on the kernel and the image in the functional in an implicit manner.

There have been several ideas for enforcing non-negativity in image processing [15, 109, 181, 13]. One such example was given by [13] who proposed a model for image reconstruction using non-negative constraints for astronomical imaging by minimising a regularised Poisson likelihood functional while the idea of back projection is similarly used in [15, 181]. The case of a Tikhonov regularisation (a much simper regulariser

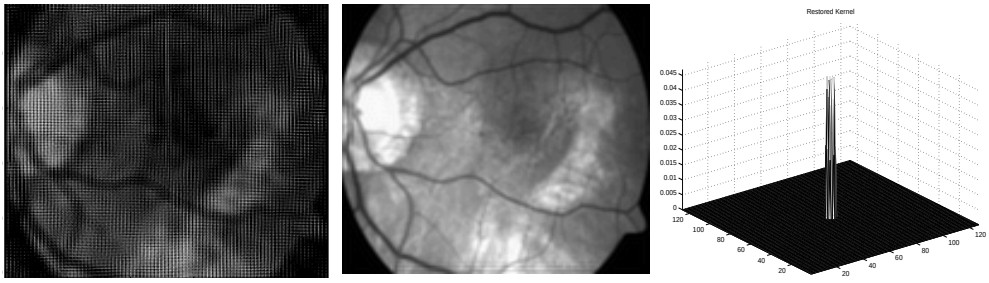


(a) Blurred Image (b) True data (c) Alg 1 Result

Figure 6.1: Good restoration results for Example 1 (box-triangle image): (b) from a corrupted image (a) using Algorithm 6. This model is able to improve the edges of the restored image (c), though the restoration is not excellent.



(a) Ex2 True Image (b) Ex2 Received Data (c) Failed restoration using Alg 6



(d) Restored u with $\kappa = 10^{-2}$ (e) Restored u with $\kappa = 1/3$ (f) Restored h with $\kappa = 1/3$

Figure 6.2: Illustration of the failure of Algorithm 6 for a retinal scan (Example 2) in (a): (b) corrupted image by motion blur; (c) failed restoration u; (d) restored u with thresholding $\kappa = 10^{-2}$; (e) restored u with thresholding $\kappa = 1/3$; (f) restored h with thresholding $\kappa = 1/3$.

than what we use here) was considered in [64]. The method of [109] ensured a positive kernel h by considering a parametric model and optimising a scalar σ which is the standard deviation.

A more sophisticated idea by Biraud [19] is to use the transform $u(x) = (E(x))^2$, with $x \in [0, 1]$, in restoring a one dimensional signal u from the model $z(x) = h(x) * u(x)+\eta(x)$, where $h(x)$ is the *known* blur function and η the noise, or $\hat{z}(s) = \hat{h}(s)\hat{u}(s)+\hat{\eta}(s)$ after Fourier transform. The central idea here is that any $E(x)$ or its Fourier transform $\hat{E}$ can lead to non-negative restoration u. For $s \in [0, s_0]$ with some cut-off frequency s_0, $\hat{h}(s) \neq 0$ so $\hat{u}(s) = \hat{z}(s)/\hat{h}(s) - \hat{\eta}(s)/\hat{h}(s)$. Noting that $u(x) = (E(x))^2$ leads to $\hat{u}(s) = \hat{E}(s) * \hat{E}(s)$, the method of Biraud [19] is

$$\hat{u}(s) = \arg\min_f \frac{1}{s_0} \int_0^{s_0} |f * f - \hat{z}(s)/\hat{h}(s)|^2 \, \mathrm{d}s. \tag{6.5}$$

To solve (6.5), a parametric iterative approach is proposed by $f_n = f_{n-1}+\bar{\omega}_n$ for $n \geq 1$. See [19]. Once a good approximation $\hat{E}(s)$ is obtained, an inverse transform would yield $E(x)$ and then a non-negative restoration $u(x)$.

Following the work of Biraud [19], Miura [138] considered generalising it to the image case and more importantly to the blind deconvolution problem by imposing non-negativity for both the image u and the blur function h. Starting from the forward problem i.e. $z(x, y) = h(x, y) * u(x, y) + \eta$, with both h and u as unknowns, he defined $h(x, y) = (\phi(x, y))^2, u(x, y) = (\psi(x, y))^2$. Then after Fourier transforms, one has

$$\hat{z}(s, t) = \{\hat{\phi}(s, t) * \hat{\phi}(s, t)\}\{\hat{\psi}(s, t) * \hat{\psi}(s, t)\} + \hat{\eta}(s, t).$$

Further similar to Biraud's method, it is proposed to solve

$$\min_{\hat{\phi},\hat{\psi}} \sum_{\hat{\phi}} \sum_{\hat{\psi}} \left| \{\hat{\phi}(s, t) * \hat{\phi}(s, t)\}\{\hat{\psi}(s, t) * \hat{\psi}(s, t)\} - \hat{z}(s, t) \right|^2$$

where the summations imply formulations after discretisation [138]. Furthermore, a conjugate gradient type solver is utilised to compute $\hat{\phi}$ and $\hat{\psi}$ which will be used to yield the non-negative solutions h and u.

To improve on the method of [54], in particular the algorithm for solving the Euler-Lagrange equations, an interesting idea was proposed in [142] to decouple the equations so that edge information of the restoration is ensured. Precisely u in the Chan-Wong Euler Lagrange equation for minimising the functional with respect to the blur function is replaced by a reference image u_τ which is obtained by using a shock filter to capture image edges in the blurred z. Then the Euler-Lagrange equations become

$$\begin{cases} u_\tau(-x, -y) * ((u_\tau * h)(x, y) - z(x, y)) - \alpha_2 \nabla \cdot \left(\frac{\nabla h(x,y)}{|\nabla h(x,y)|_\beta} \right) = 0, \\ h(-x, -y) * ((h * u)(x, y) - z(x, y)) - \alpha_1 \nabla \cdot \left(\frac{\nabla u(x,y)}{|\nabla u(x,y)|_\beta} \right) = 0, \end{cases}$$

which is a decoupled system and can be solved directly in a non-iterative way between

h and u.

In order to overcome the poor performance of [54], it is suggested in [185] that there may be a better chance of restoring the blurred image u if K blurred images $z_1, \ldots, z_K$ of the same object are available which is readily possible for video images in some situations. The minimisation proposed is

$$\min_{u,\{h_k\}} \left\{ \frac{\gamma}{2} \sum_{k=1}^{K} \|h_k * u - z_k\|^2 + Q(u) + R\left(\{h_k\}\right) \right\}$$

where in a discrete setting

$$R(\boldsymbol{h}) = \frac{\delta}{2} \boldsymbol{h}^\top \boldsymbol{R}_\Delta \boldsymbol{h} + \Psi(\boldsymbol{h}),\ \Psi(\boldsymbol{h}) = \sum \psi\left(h_k(i)\right),\ \psi(t) = \begin{cases} t & \text{if } t \geq 0 \\ +\infty & \text{otherwise} \end{cases}$$

and γ is a parameter. Here Q denotes the total variation regulariser for u and the crucial choice of $\psi(t)$ ensures positivity of h_k. However the same treatment was not applied to u. The optimisation problem was further solved by a splitting idea in an augmented Lagrangian method (ALM).

6.3 A Refined Blind Model

We now consider the single image blind deconvolution problem and propose a way to improve the algorithm by Chan and Wong [54], through use of a related and different ideas from [19, 138]. The similarity to [19, 138] lies in that, instead of treating negative components directly as in a projection method, we seek a transform that converts the original model into a new one that can satisfy the non-negativity constraints. There are three clear differences: (i) we use a different transform from previous choices; (ii) we apply regularisation to the restored quantities while previous work use non-linear least squares fitting without regularisation; (iii) we solve for u, h directly instead of solving for $\hat{u}, \hat{h}$ in the Fourier domain.

6.3.1 Choice of Positivity Transforms

We aim to impose non-negativity in the functional by representing the kernel and the image as transformed quantities which do not permit negative values. One such idea might be to represent the image as the exponential function, i.e. $u = \exp(\psi(x, y))$ for some function ψ. Unfortunately this particular transform does not work as it is not capable of dealing dark regions (where $u \approx 0$) in a stable way. In Chapter 5, we introduced a transform function which was suitable for image restoration and took the form

$$u = T(\psi) = \frac{w + 2a}{1 + e^{-\frac{2\psi}{\epsilon}}} - a, \quad \text{with } \psi = T^{-1}(u) = \frac{\epsilon}{2} \ln \frac{a + u}{a + w - u} \tag{6.6}$$

where constants $w, a, \epsilon > 0$, $\max u \leq w$ and ϵ is a small tuning parameter which controls the spread of the function. Since $T(\psi)$ for $\psi \in \mathbb{R}$ and $T^{-1}(u)$ for $u \in (-a, \infty)$

are monotone functions, we can work out their lower and upper bounds:

$$
\begin{aligned}
&u = T(\psi) \in (-a, w+a) && \text{for } \psi \in \mathbb{R};\\
&u = T(\psi) \in [0, w+a), && \text{if } \psi \geq \frac{\epsilon}{2}\ln\frac{a}{a+w};\\
&\psi = T^{-1}(u) \in \Big[\frac{\epsilon}{2}\ln\frac{a}{a+w}, \frac{\epsilon}{2}\ln\frac{a+w}{a}\Big] && \text{if } u \in [0, w] \subset (-a, w+a);\\
&\psi = T^{-1}(u) \in \Big[\frac{\epsilon}{2}\ln\frac{3a}{a+2w}, \frac{\epsilon}{2}\ln\frac{a+2w}{3a}\Big] && \text{if } u \in [-\frac{a}{2}, w+\frac{a}{2}] \subset (-a, w+a).
\end{aligned}
$$

Clearly $u = 0$ poses no problems to the transform. For instance, for the usual intensity range $[0, 255]$ for u, we may take $w = 255, a = 1/10, \epsilon = 1/100$ and note $\frac{\epsilon}{2}\ln\frac{a+w}{a} = \ln(2551)/200 \approx 0.039$ and $\frac{\epsilon}{2}\ln\frac{a+2w}{3a} = \ln(5103)/200 \approx 0.043$.

6.3.2 Reformulation of the Blind Deblurring Model

In order to apply the same transform to both the image u and the kernel h, we introduce the 8 parameters with subscripts as follows

$$
u = T_{\boldsymbol{a}}(\psi) = \frac{a_1 + 2a_4}{1 + a_2 e^{-\frac{2\psi}{a_3}}} - a_4, \qquad h = T_{\boldsymbol{b}}(\omega) = \frac{b_1 + 2b_4}{1 + b_2 e^{-\frac{2\omega}{b_3}}} - b_4,
$$

for the image and kernel respectively; here all constants can be fixed before proceeding[1]. In particular, a_1 and b_1 are the expected upper limits of the image intensity values and kernel values, a_2 and b_2 are introduced to control the values of the image and kernel at $\psi = 0$ and $\omega = 0$, and a_3 and b_3 control the spread of ψ and ω. To give one feasible set, for image u, we have either $\boldsymbol{a} = (a_1, a_2, a_3, a_4) = (255, 1, 117.16, 1)$ if $u \in [0, 255]$ or $\boldsymbol{a} = (1, 1, 0.46, 0.0039)$ if $u \in [0, 1]$, and for kernel h we take $\boldsymbol{b} = \big(1, 1, 0.46, 10^{-4}\big)$ if $h \in [0, 1]$.

We now reformulate our old problem (6.3) as the new variational model

$$
\min_{\psi,\omega} f_T(\psi, \omega) = \frac{1}{2}||T_{\boldsymbol{b}}(\omega) * T_{\boldsymbol{a}}(\psi) - z||^2_{L^2(\Omega)} + \alpha_1 \int_\Omega |\nabla T_{\boldsymbol{a}}(\psi)| + \alpha_2 \int_\Omega |\nabla T_{\boldsymbol{b}}(\omega)|, \quad (6.7)
$$

letting the image and the kernel be represented by $u = T_{\boldsymbol{a}}(\psi)$ and $h = T_{\boldsymbol{b}}(\omega)$ respectively. Here from solving equation (6.7), the non-negativity constraints are exactly and implicitly enforced i.e. $u, h \geq 0$, but the remaining symmetry and unit integral constraints on the kernel are still required.

6.4 Solution of Non-Linear Deconvolution Equations for Model (6.7)

The advantage of realising positivity is accompanied by a new challenge (or disadvantage) of having to deal with a non-linear convolution kernel in (6.7). Below we present

[1] See Appendix 5.A for more details.

two methods for solving the model and take the solution of ψ to illustrate the idea as the solution for ω is similar.

6.4.1 A Fixed Point Method

Since we have replaced the functions u and h by non-linear functions, we deal this non-linearity directly. We give a linear approximation of the transform by considering the Taylor expansion given by $T_{\boldsymbol{a}}(\psi) = A_1 + B_1\psi + O(\psi^2)$ to design an iterative scheme. First we split the transform by

$$T_{\boldsymbol{a}}(\psi) = B_1\psi + R_1(\psi)$$

where the residual $R_1(\psi)$ is non-linear, given by $R_1 = T_{\boldsymbol{a}}(\psi) - B_1\psi$. Second we employ a fixed-point lagging technique by substituting $T_{\boldsymbol{a}}(\psi, \tilde{\psi}) = B_1\psi + R_1(\tilde{\psi})$, lagging $\tilde{\psi}$ by one iteration and updating until $||\psi - \tilde{\psi}||^2_{L^2(\Omega)}$ is sufficiently small. Similarly we also have $T_{\boldsymbol{b}}(\omega, \tilde{\omega}) = B_2\omega + R_2(\tilde{\omega})$ for h.

Next we substitute these into equation (6.7) and get it linearised, leaving the remaining non-linearity in terms with known quantities. This yields the functional given by

$$\frac{1}{2}\left\|[B_2\omega + R_2(\tilde{\omega})] * [B_1\psi + R_1(\tilde{\psi})] - z\right\|^2_{L^2(\Omega)} + \alpha_1 \int_\Omega \left|\nabla T_{\boldsymbol{a}}(\psi)\right| + \alpha_2 \int_\Omega \left|\nabla T_{\boldsymbol{b}}(\omega)\right| \quad (6.8)$$

or in a form highlighting the linear terms

$$f_{TL}(\psi, \omega) = \frac{1}{2}\left\|B_1B_2\omega * \psi + B_1R_2(\tilde{\omega}) * \psi + B_2R_1(\tilde{\psi}) * \omega - \bar{z}\big(\tilde{\omega}, \tilde{\psi}\big)\right\|^2_{L^2(\Omega)}$$

$$+\alpha_1 \int_\Omega \left|\nabla T_{\boldsymbol{a}}(\psi)\right| + \alpha_2 \int_\Omega \left|\nabla T_{\boldsymbol{b}}(\omega)\right|$$

where $\bar{z}\big(\tilde{\omega}, \tilde{\psi}\big) = z - R_1(\tilde{\psi}) * R_2(\tilde{\omega})$, $B_1 = \frac{\partial}{\partial\psi} T_{\boldsymbol{a}}(\psi)\Big|_{\psi=0}$, $B_2 = \frac{\partial}{\partial\omega} T_{\boldsymbol{b}}(\omega)\big|_{\omega=0}$, $R_1\left(\tilde{\psi}\right) = T_{\boldsymbol{a}}(\psi) - B_1\psi$, $R_2\left(\tilde{\omega}\right) = T_{\boldsymbol{b}}(\omega) - B_2\omega$. Here a key observation is that residual functions R_1, R_2 are lagged in iterations, not approximated in any way.

We now wish to minimise the above functional (6.8) with respect to ψ and ω, given $\tilde{\psi}$ and $\tilde{\omega}$, thereby obtaining the Euler Lagrange equations[2]

$$B_1T_{\boldsymbol{b}}\big(\tilde{\omega}\big) * \Big(T_{\boldsymbol{b}}\big(\tilde{\omega}\big) * T_{\boldsymbol{a}}\big(\psi, \tilde{\psi}\big) - z\Big) + \alpha_1 L_1\big(\tilde{\psi}\big)\psi = 0, \quad (6.9)$$

$$B_2T_{\boldsymbol{a}}\big(\tilde{\psi}\big) * \Big(T_{\boldsymbol{a}}\big(\tilde{\psi}\big) * T_{\boldsymbol{b}}\big(\omega, \tilde{\omega}\big) - z\Big) + \alpha_2 L_2\big(\tilde{\omega}\big)\omega = 0 \quad (6.10)$$

or the linearised equations in a fixed point form

$$\begin{cases} B_1T_{\boldsymbol{b}}\left(\tilde{\omega}\right) * \Big(T_{\boldsymbol{b}}\left(\tilde{\omega}\right) * \Big(B_1\psi + R_1\big(\tilde{\psi}\big)\Big) - z\Big) + \alpha_1 L_1\big(\tilde{\psi}\big)\psi & = 0, \\ B_2T_{\boldsymbol{a}}\big(\tilde{\psi}\big) * \Big(T_{\boldsymbol{a}}\big(\tilde{\psi}\big) * \big(B_2\omega + R_2\big(\tilde{\omega}\big)\big) - z\Big) + \alpha_2 L_2\big(\tilde{\omega}\big)\omega & = 0, \end{cases}$$

[2]See Appendix 6.A for the derivation.

simplified as

$$B_1^2 T_{\boldsymbol{b}}\left(\tilde{\omega}\right) * T_{\boldsymbol{b}}\big(\tilde{\omega}\big) * \psi + \alpha_1 L_1\big(\tilde{\psi}\big)\psi = B_1 \bar{z}_1, \tag{6.11}$$

$$B_2^2 T_{\boldsymbol{a}}\big(\tilde{\psi}\big) * T_{\boldsymbol{a}}\big(\tilde{\psi}\big) * \omega + \alpha_2 L_2\left(\tilde{\omega}\right)\omega = B_2 \bar{z}_2, \tag{6.12}$$

where

$$\bar{z}_1 = T_{\boldsymbol{b}}\big(\tilde{\omega}\big) * \Big(z - T_{\boldsymbol{b}}\big(\tilde{\omega}\big) * R_1\big(\tilde{\psi}\big)\Big),$$
$$\bar{z}_2 = T_{\boldsymbol{a}}\big(\tilde{\psi}\big) * \Big(z - T_{\boldsymbol{a}}\big(\tilde{\psi}\big) * R_2\big(\tilde{\omega}\big)\Big),$$

$$L_1\big(\tilde{\psi}\big)\psi = \frac{4\tilde{E}_1(a_1+2a_4)(\tilde{E}_1-1)|\nabla\psi|}{(1+\tilde{E}_1)^3 a_3^2} - \nabla\cdot\left(\frac{2(a_1+2a_4)\tilde{E}_1}{(1+\tilde{E}_1)^2 a_3|\nabla\tilde{\psi}|}\nabla\psi\right),$$

$$L_2\left(\tilde{\omega}\right)\omega = \frac{4\tilde{E}_2(b_1+2b_4)(\tilde{E}_2-1)|\nabla\omega|}{(1+\tilde{E}_2)^3 b_3^2} - \nabla\cdot\left(\frac{2(b_1+2b_4)\tilde{E}_2}{(1+\tilde{E}_2)^2 b_3|\nabla\tilde{\omega}|}\nabla\omega\right),$$

$$\tilde{E}_1 = E_1\big(\tilde{\psi}\big) = a_2 \exp\left(\frac{-2\tilde{\psi}}{a_3}\right), \qquad \tilde{E}_2 = E_2\left(\tilde{\omega}\right) = b_2 \exp\left(\frac{-2\tilde{\omega}}{b_3}\right).$$

After making the initial estimates $u^{(0)}$ and $h^{(0)}$ of the image and the kernel respectively, we apply the inverse transforms obtaining $\psi^{(0)}$ and $\omega^{(0)}$. We then solve equations (6.9) and (6.10) as part of an alternate minimisation scheme.

6.4.2 Boundary conditions

We adopt the Dirichlet boundary conditions $u\big|_\Gamma = h\big|_\Gamma = 0$ which become $\psi\big|_\Gamma = -a_3/2 \ln\left(\frac{a_1+a_4}{a_4 a_2}\right)$ and $\omega\big|_\Gamma = -b_3/2 \ln\left(\frac{b_1+b_4}{b_4 b_2}\right)$. However, we can choose a_2, b_2 such that $\frac{a_1+a_4}{a_4 a_2} = \frac{b_1+b_4}{b_4 b_2} = 1$ i.e. $\psi\big|_\Gamma = \omega\big|_\Gamma = 0$.

6.4.3 Kernel Constraints

Note that the previously mentioned constraints $h(x,y) = h(-x,-y)$ and $\int_\Omega h(x,y)\,\mathrm{d}\Omega = 1$ take the new forms: $T_{\boldsymbol{b}}\left(\omega(x,y)\right) = T_{\boldsymbol{b}}\left(\omega(-x,-y)\right)$ and $\int_\Omega T_{\boldsymbol{b}}\left(\omega(x,y)\right)\mathrm{d}\Omega = 1$. We can satisfy the first condition by imposing

$$\omega = \frac{1}{2}\left(\hat{\omega}(x,y) + \hat{\omega}(-x,-y)\right)$$

where $\tilde{\omega}$ is the result of the previous step. For the second constraint in the discrete setting, we interpret the integral of a function over the domain Ω as the sum of all

values of the function in Ω. Letting $\hat{S} = \sum_x \sum_y T_{\boldsymbol{b}}(\hat{\omega})$, our constraint is given by

$$\begin{aligned} T_{\boldsymbol{b}}(\omega) = \hat{S}^{-1} T_{\boldsymbol{b}}(\hat{\omega}) \quad &\Longleftrightarrow \quad \frac{b_1 + 2b_4}{1 + b_2 e^{\frac{-2\omega}{b_3}}} - b_4 = \hat{S}^{-1} \left(\frac{b_1 + 2b_4}{1 + b_2 e^{\frac{-2\hat{\omega}}{b_3}}} - b_4 \right) \\ &\Longleftrightarrow \quad \hat{S} \frac{b_1 + 2b_4}{1 + b_2 e^{\frac{-2\omega}{b_3}}} - \hat{S} b_4 = \frac{b_1 + 2b_4}{1 + b_2 e^{\frac{-2\hat{\omega}}{b_3}}} - b_4 \\ &\Longleftrightarrow \quad \hat{S} \frac{b_1 + 2b_4}{1 + b_2 e^{\frac{-2\omega}{b_3}}} = \frac{b_1 + 2b_4}{1 + b_2 e^{\frac{-2\hat{\omega}}{b_3}}} + (\hat{S} - 1) b_4 \end{aligned}$$

$$\begin{aligned} &\Longleftrightarrow \quad \hat{S} \frac{b_1 + 2b_4}{1 + b_2 e^{\frac{-2\omega}{b_3}}} = \frac{b_1 + 2b_4 + (\hat{S} - 1) b_4 \left(1 + b_2 e^{\frac{-2\hat{\omega}}{b_3}} \right)}{1 + b_2 e^{\frac{-2\hat{\omega}}{b_3}}} \\ &\Longleftrightarrow \quad 1 + b_2 e^{\frac{-2\omega}{b_3}} = \hat{S} \frac{(b_1 + 2b_4) \left(1 + b_2 e^{\frac{-2\hat{\omega}}{b_3}} \right)}{b_1 + 2b_4 + (\hat{S} - 1) b_4 \left(1 + b_2 e^{\frac{-2\hat{\omega}}{b_3}} \right)} \\ &\Longleftrightarrow \quad \omega = \frac{-b_3}{2} \ln \left(\frac{\hat{S}}{b_2} \frac{(b_1 + 2b_4) \left(1 + b_2 e^{\frac{-2\hat{\omega}}{b_3}} \right)}{b_1 + 2b_4 + (\hat{S} - 1) b_4 \left(1 + b_2 e^{\frac{-2\hat{\omega}}{b_3}} \right)} - \frac{1}{b_2} \right). \end{aligned}$$

The solution method is given in Algorithm 17 below.

Algorithm 17 The First Transform Method (TM1) for Model (6.7) via (6.9) and (6.10).

function $(h, u) \leftarrow$TM1$(h^{(0)}, u^{(0)}, z, \alpha_1, \alpha_2, maxit, tol)$
  $\omega^{(0)} = T_{\boldsymbol{b}}^{-1}\left(h^{(0)}\right)$
  $\psi^{(0)} = T_{\boldsymbol{a}}^{-1}\left(u^{(0)}\right)$
  for $k \leftarrow 1$ **to** $maxit$ **do**
    $\omega^{(k+\frac{1}{3})} \leftarrow$ SOLVE (6.10)
    $\omega^{(k+\frac{2}{3})} \leftarrow \frac{1}{2}\left(\omega^{(k+\frac{1}{3})}(x, y) + \omega^{(k+\frac{1}{3})}(-x, -y)\right)$
    $S \leftarrow \sum_x \sum_y T_{\boldsymbol{b}}\left(\omega^{(k+\frac{2}{3})}\right)$
    $\omega^{(k+1)} = \frac{-b_3}{2} \ln \left(\frac{S}{b_2} \frac{(b_1+2b_4)\left(1+b_2 e^{\frac{-2\omega^{(k+\frac{2}{3})}}{b_3}}\right)}{b_1+2b_4+(S-1)b_4\left(1+b_2 e^{\frac{-2\omega^{(k+\frac{2}{3})}}{b_3}}\right)} - \frac{1}{b_2} \right)$
    $\psi^{(k+1)} \leftarrow$ SOLVE (6.9).
    If $\|\psi^{(k+1)} - \psi^{(k)}\| + \|\omega^{(k+1)} - \omega^{(k)}\| \leq tol$, then exit or continue.
  end for
  Accept the restored image $u = T_{\boldsymbol{a}}(\psi)$ and the restored kernel $h = T_{\boldsymbol{b}}(\omega)$.
end function

6.4.4 Numerical Implementation

We shall briefly discuss the discretisation of the linearised operators in (6.9) and (6.10) by a finite difference method.

Discretisation of the fitting (integral) term

We wish to discretise the quantity $\omega * \psi$. As it is the usual convolution operator and its discretisation under discussion, it is useful to consider temporarily the quantity $h * u$, related to $\left(h^T(hu - z)\right)$. We begin by finding a matrix H such that $Hu = h * u$. The matrix H can be shown to have a Block-Toeplitz-with-Toeplitz-Blocks (BTTB) structure [147, 146] if the zero Dirichlet boundary condition is assumed. We can calculate H as

$$H = \begin{pmatrix} H_{-m+1,-m+1} & H_{-m+1,-m+2} & \cdots & H_{-m+1,m+n} \\ H_{-m+2,-m+1} & \ddots & \ddots & \ddots \\ \vdots & \ddots & \ddots & \ddots \\ H_{m+n,-m+1} & \ddots & \ddots & \ddots \end{pmatrix}$$

$$H_{a,b} = \begin{pmatrix} h_{0,a-b} & \cdots & h_{-2m-n+1,a-b} \\ \vdots & \ddots & \ddots \\ h_{2m+n-1,a-b} & \ddots & \ddots \end{pmatrix}$$

where n is the size of our grid and m is the number of points outside of the image to be included for the consideration of boundary points. Writing an extended version of our image which is "padded" to include points outside of the image as

$$u_{ext} = \begin{pmatrix} \boxed{\begin{matrix} u_{1-m,1-m} & \cdots & u_{1-m,n-m} \\ \vdots & & \vdots \\ u_{n-m,1-m} & \cdots & u_{n-m,n-m} \end{matrix}} & \begin{matrix} u_{1-m,n-m+1} & \cdots & u_{1-m,n+m} \\ \vdots & & \vdots \\ u_{n-m,n-m+1} & \cdots & u_{n-m,n+m} \end{matrix} \\ \begin{matrix} u_{n-m+1,1-m} & \cdots & u_{n-m+1,n-m} \\ \vdots & & \vdots \\ u_{n+m,1-m} & \cdots & u_{n+m,n-m} \end{matrix} & \begin{matrix} u_{n-m+1,n-m+1} & \cdots & u_{n-m+1,n+m} \\ \vdots & & \vdots \\ u_{n+m,n-m+1} & \cdots & u_{n+m,n+m} \end{matrix} \end{pmatrix}$$

where the image is contained within the square and the points outside are points outside of the image,[3] we split the extended image into sub-regions defined by

$$\begin{aligned} \Omega_1 &= \left\{(i,j) \;\middle|\; 1-m \le i \le n-m \text{ and } 1-m \le j \le n-m\right\}, \\ \Omega_2 &= \left\{(i,j) \;\middle|\; 1-m \le i \le n-m \text{ and } n-m+1 \le j \le n+m\right\}, \\ \Omega_3 &= \left\{(i,j) \;\middle|\; n-m+1 \le i \le n+m \text{ and } 1-m \le j \le n-m\right\}, \\ \Omega_4 &= \left\{(i,j) \;\middle|\; n-m+1 \le i \le n+m \text{ and } n-m+1 \le j \le n+m\right\}. \end{aligned}$$

[3]Note that here we could "pad" the image on all sides rather than to the right and bottom but for ease of notation we write it in this way.

and hence define Dirichlet boundary conditions by

$$(u_{ext})_{i,j} = \begin{cases} u_{i,j} & \text{if } (i,j) \in \Omega_1 \\ 0 & \text{if } (i,j) \in \Omega_2 \cup \Omega_3 \cup \Omega_4. \end{cases}$$

Our matrix H allowing for Dirichlet boundary conditions is therefore given by

$$H = \begin{pmatrix} H_{-m+1,-m+1} & \cdots & H_{-m+1,0} & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots \\ H_{2m+n-1,-m+1} & \ddots & \ddots & \ddots & \ddots & \ddots \\ 0 & \ddots & \ddots & \ddots & \ddots & \ddots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots \\ 0 & \ddots & \ddots & \ddots & \ddots & \ddots \end{pmatrix}$$

$$H_{a,b} = \begin{pmatrix} h_{0,a-b} & \cdots & h_{1-m,a-b} & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots \\ h_{n-m,a-b} & \ddots & \ddots & \ddots & \ddots & \ddots \\ 0 & \ddots & \ddots & \ddots & \ddots & \ddots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots \\ 0 & \ddots & \ddots & \ddots & \ddots & \ddots \end{pmatrix}.$$

Note that as well as having a BTTB structure, with suitably chosen m and assuming that the kernel function h is symmetric then the matrix H is also symmetric.

Discretisation of the Total Variation regularisation (differential) term

For the implementation of this term, we expand the gradients and evaluate the divergence as follows. If A, B, C, D are functions of x then $\left(AB(CD)^{-1}\right)_x = A_x B(CD)^{-1} + AB_x(CD)^{-1} - AB(CD)^{-2}\left(C_x D + CD_x\right)$. Letting $A = E_1$, $B = \psi_x$, $C = (1+E_1)^2$, $D = |\nabla\psi|$, we have

$$A_x = \frac{-2\psi_x E_1}{a_3}, \quad B_x = \psi_{xx}, \quad C_x = -4\psi_x E_1 \frac{1+E_1}{a_3} \quad \text{and} \quad D_x = \frac{\psi_x\psi_{xx} + \psi_y\psi_{xy}}{|\nabla\psi|}.$$

We thus give the expansion as

$$\left(\frac{E_1\psi_x}{(1+E_1)^2|\nabla\psi|}\right)_x = \frac{-2\psi_x^2 E_1 + E_1\psi_{xx}}{a_3(1+E_1)^2|\nabla\psi|} + \frac{4E_1^2\psi_x^2}{a_3(1+E_1)^3|\nabla\psi|} - \frac{E_1\psi_x\left(\psi_x\psi_{xx} + \psi_y\psi_{xy}\right)}{(1+E_1)^4|\nabla\psi|^2}, \tag{6.13}$$

$$\left(\frac{E_1\psi_y}{(1+E_1)^2|\nabla\psi|}\right)_y = \frac{-2\psi_y^2 E_1 + E_1\psi_{yy}}{a_3(1+E_1)^2|\nabla\psi|} + \frac{4E_1^2\psi_y^2}{a_3(1+E_1)^3|\nabla\psi|} - \frac{E_1\psi_y\left(\psi_y\psi_{yy} + \psi_x\psi_{yx}\right)}{(1+E_1)^4|\nabla\psi|^2}. \tag{6.14}$$

We can then give the expansion as

$$\begin{aligned}
&\frac{4\tilde{E}_1(a_1+2a_4)(\tilde{E}_1-1)|\nabla\psi|}{(1+\tilde{E}_1)^3a_3^2}-\nabla\cdot\left(\frac{2(a_1+2a_4)\tilde{E}_1}{(1+\tilde{E}_1)^2a_3|\nabla\tilde{\psi}|}\nabla\psi\right)\\
&= \frac{4E_1(a_1+2a_4)(E_1-1)|\nabla\psi|}{(1+E_1)^3a_3^2}-\frac{E_1\left(\Delta\psi-2|\nabla\psi|^2\right)}{a_3\left(1+E_1\right)^2|\nabla\psi|}\\
&\quad+\frac{4E_1^2|\nabla\psi|}{a_3\left(1+E_1\right)^3}-\frac{E_1\left(\psi_x^2\psi_{xx}+\psi_y^2\psi_{yy}+2\psi_x\psi_y\psi_{xy}\right)}{\left(1+E_1\right)^4|\nabla\psi|^2}.
\end{aligned}$$

In order to implement our model in image deblurring, we define an $n\times n$ grid Ω and a function $u:\Omega\to\mathbb{R}$ which maps each point on the grid to an intensity value of the image, thereby obtaining an $n\times n$ matrix u. We approximate derivatives using finite differences, derived from the definition $u_x=\partial u/\partial x=\lim_{\epsilon\to 0}\epsilon^{-1}(u(x+\epsilon,y)-u(x))$ and defined as

$$u_x(i,j)=\frac{u(i+1,j)-u(i,j)}{h}=\frac{u(i+1/2,j)-u(i-1/2,j)}{h}=\frac{u(i,j)-u(i-1,j)}{h}$$

for forward, central and backward differences respectively where h is the step size. We similarly define the derivative with respect to y as

$$u_y(i,j)=\frac{u(i,j+1)-u(i,j)}{h}=\frac{u(i,+1/2)-u(i,j-1/2)}{h}=\frac{u(i,j)-u(i,j-1)}{h}$$

where h is the step size. Using this, we may discretise our model and build the Total Variation transform terms.

We can then give the discretisation at a particular point as

$$\begin{aligned}
(TV(T_{\boldsymbol{a}}(\psi)))_{i,j} &= \frac{4\left(E_1\right)_{i,j}(a_1+2a_4)(\left(E_1\right)_{i,j}-1)|\nabla\psi_{i,j}|}{(1+\left(E_1\right)_{i,j})^3a_3^2}+\frac{4\left(E_1\right)_{i,j}^2|\nabla\psi_{i,j}|}{a_3\left(1+\left(E_1\right)_{i,j}\right)^3}\\
&\quad+\frac{\left(E_1\right)_{i,j}\left(4\psi_{i,j}\left(1+\psi_{i,j}-\psi_{i+1,j}-\psi_{i,j+1}\right)\right)}{a_3h^2\left(1+\left(E_1\right)_{i,j}\right)^2|\nabla\psi|_{i,j}}\\
&\quad+\frac{\left(E_1\right)_{i,j}\left(\psi_{i+1,j}\left(\psi_{i+1,j}-1\right)+\psi_{i,j+1}\left(\psi_{i,j+1}-1\right)\right)}{a_3h^2\left(1+\left(E_1\right)_{i,j}\right)^2|\nabla\psi|_{i,j}}\\
&\quad-\frac{\left(E_1\right)_{i,j}\left(\psi_{i-1,j}+\psi_{i,j-1}\right)}{a_3h^2\left(1+\left(E_1\right)_{i,j}\right)^2|\nabla\psi|_{i,j}}\\
&\quad-\frac{\left(E_1\right)_{i,j}\left(\Psi_1\psi_{i,j}+\Psi_2\psi_{i+1,j}+\Psi_3\psi_{i,j+1}+\Psi_4\psi_{i+1,j+1}\right)}{\left(1+\left(E_1\right)_{i,j}\right)^4|\nabla\psi_{i,j}|^2}
\end{aligned}$$

where

$$\begin{aligned}
|\nabla\psi_{i,j}| &= \tfrac{1}{h}\sqrt{2\psi_{i,j}\left(\psi_{i,j}-\psi_{i+1,j}-\psi_{i,j+1}\right)+\psi_{i+1,j}^2+\psi_{i,j+1}^2},\\
|\nabla\psi_{i,j}|^2 &= \tfrac{2\psi_{i,j}}{h^2}\left(\psi_{i,j}-\psi_{i+1,j}-\psi_{i,j+1}\right)+\tfrac{\psi_{i+1,j}^2+\psi_{i,j+1}^2}{h^2},
\end{aligned}$$

$$
\begin{aligned}
(E_1)_{i,j} &= e^{\frac{-2\psi_{i,j}}{a_3}}, \\
\Psi_1 &= \psi_{i,j}\psi_{i-1,j} - 2\psi_{i,j}^2 + \psi_{i,j}\psi_{i+1,j} - 2\psi_{i+1,j}\psi_{i-1,j} - 2\psi_{i+1,j}^2 \\
&\quad + \psi_{i,j}\psi_{i,j-1} + \psi_{i,j}\psi_{i,j+1} - 2\psi_{i,j-1}\psi_{i,j+1} - 2\psi_{i,j+1}^2 \\
&\quad + 2\psi_{i,j}\psi_{i+1,j+1} + 6\psi_{i+1,j}\psi_{i,j+1} - 2\psi_{i+1,j}\psi_{i+1,j+1} - \psi_{i,j+1}\psi_{i+1,j+1}, \\
\Psi_2 &= \psi_{i+1,j}\left(\psi_{i-1,j} + \psi_{i+1,j}\right) - 2\psi_{i,j+1}^2, \\
\Psi_3 &= \psi_{i,j+1}\left(\psi_{i,j-1} + \psi_{i,j+1}\right) - 2\psi_{i+1,j}^2, \\
\Psi_4 &= 2\psi_{i+1,j}\psi_{i,j+1}.
\end{aligned}
$$

From the above equation, we can clearly see that we have some unknown values on the boundary, that is when $i = 1, n$ and $j = 1, n$. At these points, we must make some assumption about the unknown values. If we assume that Dirichlet boundary conditions are appropriate then we assume that the image intensity value is equal to a constant c outside of the image. This is typically given as 0 but we will keep the general case. Recalling that our 'image' ψ is the transform of the image u, we let our Dirichlet boundary condition be given by $\psi_{i,j} = 0$ if $i, j \notin [1, n]$. To give an example, we will show the equation at the boundary when $i = 1$ and $i = n$. The case for $j = 1, n$ is similar. At the left boundary $i = 1$, for all values of j, we have

$$
\begin{aligned}
(TV(T_{\boldsymbol{a}}(\psi)))_{1,j} &= \frac{4\left(E_1\right)_{1,j}\left(a_1 + 2a_4\right)\left(\left(E_1\right)_{1,j} - 1\right)|\nabla\psi_{1,j}|}{\left(1 + \left(E_1\right)_{1,j}\right)^3 a_3^2} + \frac{4\left(E_1\right)_{1,j}^2|\nabla\psi_{1,j}|}{a_3\left(1 + \left(E_1\right)_{1,j}\right)^3} \\
&\quad + \frac{\left(E_1\right)_{1,j}\left(4\psi_{1,j}\left(1 + \psi_{1,j} - \psi_{2,j} - \psi_{1,j+1}\right)\right)}{a_3 h^2\left(1 + \left(E_1\right)_{1,j}\right)^2|\nabla\psi|_{1,j}} \\
&\quad + \frac{\left(E_1\right)_{1,j}\left(\psi_{2,j}\left(\psi_{2,j} - 1\right) + \psi_{1,j+1}\left(\psi_{1,j+1} - 1\right)\right)}{a_3 h^2\left(1 + \left(E_1\right)_{1,j}\right)^2|\nabla\psi|_{1,j}} \\
&\quad - \frac{\left(E_1\right)_{1,j}\left(c_T + \psi_{1,j-1}\right)}{a_3 h^2\left(1 + \left(E_1\right)_{1,j}\right)^2|\nabla\psi|_{1,j}} \\
&\quad - \frac{\left(E_1\right)_{1,j}\left(\Psi_1\psi_{1,j} + \Psi_2\psi_{2,j} + \Psi_3\psi_{1,j+1} + \Psi_4\psi_{2,j+1}\right)}{\left(1 + \left(E_1\right)_{1,j}\right)^4|\nabla\psi_{1,j}|^2}
\end{aligned}
$$

where

$$
\begin{aligned}
|\nabla\psi_{1,j}| &= \tfrac{1}{h}\sqrt{2\psi_{1,j}\left(\psi_{1,j} - \psi_{2,j} - \psi_{1,j+1}\right) + \psi_{2,j}^2 + \psi_{1,j+1}^2}, \\
|\nabla\psi_{1,j}|^2 &= \tfrac{2\psi_{1,j}}{h^2}\left(\psi_{1,j} - \psi_{2,j} - \psi_{1,j+1}\right) + \tfrac{\psi_{2,j}^2 + \psi_{1,j+1}^2}{h^2}, \\
\left(E_1\right)_{1,j} &= e^{\frac{-2\psi_{1,j}}{a_3}},
\end{aligned}
$$

$$
\begin{aligned}
\Psi_1 &= \psi_{1,j}c_T - 2\psi_{1,j}^2 + \psi_{1,j}\psi_{2,j} - 2\psi_{2,j}c_T - 2\psi_{2,j}^2 + \psi_{1,j}\psi_{1,j-1} + \psi_{1,j}\psi_{1,j+1} \\
&\quad - 2\psi_{1,j-1}\psi_{1,j+1} - 2\psi_{1,j+1}^2 + 2\psi_{1,j}\psi_{2,j+1} + 6\psi_{2,j}\psi_{1,j+1} - 2\psi_{2,j}\psi_{2,j+1} - \psi_{1,j+1}\psi_{2,j+1}, \\
\Psi_2 &= \psi_{2,j}\left(c_T + \psi_{2,j}\right) - 2\psi_{1,j+1}^2, \\
\Psi_3 &= \psi_{1,j+1}\left(\psi_{1,j-1} + \psi_{1,j+1}\right) - 2\psi_{2,j}^2, \\
\Psi_4 &= 2\psi_{2,j}\psi_{1,j+1}.
\end{aligned}
$$

At the right boundary $i = n$, for all values of j, we have

$$
\begin{aligned}
\left(TV(T_{\boldsymbol{a}}(\psi))\right)_{n,j} &= \frac{4\left(E_1\right)_{n,j}\left(a_1 + 2a_4\right)\left(\left(E_1\right)_{n,j} - 1\right)|\nabla\psi_{n,j}|}{\left(1 + \left(E_1\right)_{n,j}\right)^3 a_3^2} + \frac{4\left(E_1\right)_{n,j}^2|\nabla\psi_{n,j}|}{a_3\left(1 + \left(E_1\right)_{n,j}\right)^3} \\
&\quad + \frac{\left(E_1\right)_{n,j}\left(4\psi_{n,j}\left(1 + \psi_{n,j} - c_T - \psi_{n,j+1}\right)\right)}{a_3h^2\left(1 + \left(E_1\right)_{n,j}\right)^2|\nabla\psi|_{n,j}} \\
&\quad + \frac{\left(E_1\right)_{n,j}\left(c_T\left(c_T - 1\right) + \psi_{n,j+1}\left(\psi_{n,j+1} - 1\right)\right)}{a_3h^2\left(1 + \left(E_1\right)_{n,j}\right)^2|\nabla\psi|_{n,j}} \\
&\quad - \frac{\left(E_1\right)_{n,j}\left(\psi_{n-1,j} + \psi_{n,j-1}\right)}{a_3h^2\left(1 + \left(E_1\right)_{n,j}\right)^2|\nabla\psi|_{n,j}} \\
&\quad - \frac{\left(E_1\right)_{n,j}\left(\Psi_1\psi_{n,j} + \Psi_2c_T + \Psi_3\psi_{n,j+1} + \Psi_4c_T\right)}{\left(1 + \left(E_1\right)_{n,j}\right)^4|\nabla\psi_{n,j}|^2}
\end{aligned}
$$

where

$$
\begin{aligned}
|\nabla\psi_{n,j}| &= \tfrac{1}{h}\sqrt{2\psi_{n,j}\left(\psi_{n,j} - c_T - \psi_{n,j+1}\right) + c_T^2 + \psi_{n,j+1}^2}, \\
|\nabla\psi_{n,j}|^2 &= \tfrac{2\psi_{n,j}}{h^2}\left(\psi_{n,j} - c_T - \psi_{n,j+1}\right) + \tfrac{c_T^2 + \psi_{n,j+1}^2}{h^2}
\end{aligned}
$$

$$
\begin{aligned}
\left(E_1\right)_{n,j} &= e^{\frac{-2\psi_{n,j}}{a_3}}, \\
\Psi_1 &= \psi_{n,j}\psi_{n-1,j} - 2\psi_{n,j}^2 + \psi_{n,j}c_T - 2c_T\psi_{n-1,j} - 2c_T^2 + \psi_{n,j}\psi_{n,j-1} + \psi_{n,j}\psi_{n,j+1}, \\
&\quad - 2\psi_{n,j-1}\psi_{n,j+1} - 2\psi_{n,j+1}^2 + 2\psi_{n,j}c_T + 6c_T\psi_{n,j+1} - 2c_T^2 - \psi_{n,j+1}c_T, \\
\Psi_2 &= c_T\left(\psi_{n-1,j} + c_T\right) - 2\psi_{n,j+1}^2, \\
\Psi_3 &= \psi_{n,j+1}\left(\psi_{n,j-1} + \psi_{n,j+1}\right) - 2c_T^2, \\
\Psi_4 &= 2c_T\psi_{n,j+1}.
\end{aligned}
$$

We can have simplification at corners. For example, when $(i, j) = (n, n)$, we have

$$
\begin{aligned}
\left(TV(T_{\boldsymbol{a}}(\psi))\right)_{n,n} &= \frac{4\left(E_1\right)_{n,n}\left(a_1 + 2a_4\right)\left(\left(E_1\right)_{n,n} - 1\right)|\nabla\psi_{n,n}|}{\left(1 + \left(E_1\right)_{n,n}\right)^3 a_3^2} + \frac{4\left(E_1\right)_{n,n}^2|\nabla\psi_{n,n}|}{a_3\left(1 + \left(E_1\right)_{n,n}\right)^3} \\
&\quad + \frac{\left(E_1\right)_{n,n}\left(4\psi_{n,n}\left(1 + \psi_{n,n} - 2c_T\right) + 2\left(c_T - 1\right) - \left(\psi_{n-1,n} + \psi_{n,n-1}\right)\right)}{a_3h^2\left(1 + \left(E_1\right)_{n,n}\right)^2|\nabla\psi|_{n,n}} \\
&\quad - \frac{\left(E_1\right)_{n,n}\left(\Psi_1\psi_{n,n} + c_T^2\left(\psi_{n-1,n} + \psi_{n,n-1}\right)\right)}{\left(1 + \left(E_1\right)_{n,n}\right)^4|\nabla\psi_{n,n}|^2}
\end{aligned}
$$

where

$$|\nabla\psi_{n,n}| = \frac{\sqrt{2}}{h}|\psi_{n,n} - c_T|, \qquad |\nabla\psi_{n,n}|^2 = \frac{2}{h^2}\left(\psi_{n,n} - c_T\right)^2,$$

$$\Psi_1 = \psi_{n,n}\left(\psi_{n-1,n} - 2\psi_{n,n} + 4c_T + \psi_{n,n-1}\right) - c_T\left(c_T + 2\psi_{n,n-1} + 2\psi_{n-1,n}\right).$$

Given this and the matrix-vector mapping $(i,j) \mapsto (i + n(j-1))$, we now build the sparse $n^2 \times n^2$ 5-diagonal coefficient matrix C^{TV} for the transformed Total Variation such that $C^{TV}\Psi + R^{TV} = TV(T_{\boldsymbol{a}})$, which may be given by

$$\begin{aligned}
C_{i+n(j-1),i+n(j-1)} &= \frac{4(E_1)_{i,j}(1+\psi_{i,j}-\psi_{i+1,j}-\psi_{i,j+1})}{a_3h^2(1+(E_1)_{i,j})^2|\nabla\psi|_{i,j}} - \frac{(E_1)_{i,j}\Psi_1}{(1+(E_1)_{i,j})^4|\nabla\psi|^2_{i,j}},\\
C_{i+n(j-1),i+n(j-1)+1} &= \frac{(E_1)_{i,j}(\psi_{i+1,j}-1)}{a_3h^2(1+(E_1)_{i,j})^2|\nabla\psi|_{i,j}} - \frac{(E_1)_{i,j}\Psi_2}{(1+(E_1)_{i,j})^4|\nabla\psi|^2_{i,j}},\\
C_{i+n(j-1),i+nj} &= \frac{(E_1)_{i,j}(\psi_{i,j+1}-1)}{a_3h^2(1+(E_1)_{i,j})^2|\nabla\psi|_{i,j}} - \frac{(E_1)_{i,j}\Psi_3}{(1+(E_1)_{i,j})^4|\nabla\psi|^2_{i,j}},\\
C_{i+n(j-1),i+n(j-1)-1} &= \frac{-(E_1)_{i,j}}{a_3h^2(1+(E_1)_{i,j})^2|\nabla\psi|_{i,j}},\\
C_{i+n(j-1),i+n(j-2)} &= \frac{-(E_1)_{i,j}}{a_3h^2(1+(E_1)_{i,j})^2|\nabla\psi|_{i,j}},
\end{aligned}$$

$$\begin{aligned}
R^{TV}_{i+n(j-1)} &= \frac{4\left(E_1\right)_{i,j}\left(a_1+2a_4\right)\left(\left(E_1\right)_{i,j}-1\right)|\nabla\psi_{i,j}|}{(1+(E_1)_{i,j})^3a_3^2}\\
&\quad + \frac{4\left(E_1\right)^2_{i,j}|\nabla\psi_{i,j}|}{a_3\left(1+(E_1)_{i,j}\right)^3} - \frac{(E_1)_{i,j}\Psi_4\psi_{i+1,j+1}}{(1+(E_1)_{i,j})^4|\nabla\psi|^2_{i,j}}.
\end{aligned}$$

We now look at how to implement the boundary conditions in the matrices C^{TV} and R^{TV}. Assuming that we have Dirichlet boundary conditions given $\psi_{i,j} = c_T$ as defined above for all $(i,j) \notin \Omega$ then, for example, at $(i,j) = (n,n)$ we have

$$\begin{aligned}
C_{n^2,n^2} &= \frac{4(E_1)_{n,n}(1+\psi_{n,n}-2c_T)}{a_3h^2(1+(E_1)_{n,n})^2|\nabla\psi|_{n,n}} - \frac{(E_1)_{n,n}\Psi_1}{(1+(E_1)_{n,n})^4|\nabla\psi|^2_{n,n}},\\
C_{n^2,n^2-1} &= \frac{-(E_1)_{n,n}}{a_3h^2(1+(E_1)_{n,n})^2|\nabla\psi|_{n,n}}, \qquad C_{n^2,n^2-n} = \frac{-(E_1)_{n,n}}{a_3h^2(1+(E_1)_{n,n})^2|\nabla\psi|_{n,n}},\\
R^{TV}_{n^2} &= \frac{4\left(E_1\right)_{n,n}\left(a_1+2a_4\right)\left(\left(E_1\right)_{n,n}-1\right)|\nabla\psi_{n,n}|}{(1+(E_1)_{n,n})^3a_3^2}\\
&\quad + \frac{4\left(E_1\right)^2_{n,n}|\nabla\psi_{n,n}|}{a_3\left(1+(E_1)_{n,n}\right)^3} - \frac{(E_1)_{n,n}\Psi_4\psi_{n+1,n+1}}{(1+(E_1)_{n,n})^4|\nabla\psi|^2_{n,n}}.
\end{aligned}$$

Iterative solution of linear systems

We now consider a solution method for solving our system of discrete versions of linearised PDEs (6.9) and (6.10) or (6.11) and (6.12). We use a preconditioned conjugate

gradient algorithm; see [195, 60]. In order to improve the speed of convergence, we make use of preconditioners P_1 and P_2. We implement the product preconditioner following the work of [195] and given by

$$P_1 = \frac{1}{\gamma_1}\left(\tilde{v_1}^c * \tilde{v_1}^c + \gamma_1 I\right)^{\frac{1}{2}} \left(\gamma_1 I + \alpha_1 L_1\left(\tilde{\psi}\right)\right)\left(\tilde{v_1}^c * \tilde{v_1}^c + \gamma_1 I\right)^{\frac{1}{2}} \tag{6.15}$$

for equation (6.9) where $\tilde{v_1}^c$ is a circulant approximation [195, 60] to $\tilde{v_1}$ defined above, γ_1 is a positive constant and L_1 is given in (6.11). Similarly, for equation (6.10), we use the preconditioner given by

$$P_2 = \frac{1}{\gamma_2}\left(\tilde{v_2}^c * \tilde{v_2}^c + \gamma_2 I\right)^{\frac{1}{2}} \left(\gamma_2 I + \alpha_2 L_2\left(\tilde{\omega}\right)\right)\left(\tilde{v_2}^c * \tilde{v_2}^c + \gamma_2 I\right)^{\frac{1}{2}} \tag{6.16}$$

where $\tilde{v_2}^c$ is a circulant approximation to $\tilde{v_2}$ defined above, γ_2 is a positive constant and L_2 is given in (6.12).

6.4.5 A Fast Splitting Method

The central idea of our second method for model (6.7) is to remove the non-linearity in the fitting term or maintain the fitting term as linear in u and h.

This is achieved by splitting the model into separate tasks of deblurring and denoising using an alternate direction method (ADM) [44, 104, 198, 188, 197]. More importantly for our transformed formulation, the non-linearity introduced by the transforms to the blurring term is removed by the method.

Starting with the unconstrained non-negative problem given by equation (6.3) we create the augmented Lagrangian minimising functional

$$\begin{aligned} f(u,h,\psi,\omega,\lambda_1,\lambda_2) = \frac{1}{2}||h*u-z||^2_{L^2(\Omega)} + \alpha_1 L_1(T_{\boldsymbol{a}}(\psi)) + \frac{\gamma_1}{2}||u-T_{\boldsymbol{a}}(\psi)||^2_{L^2(\Omega)} \\ + <\lambda_1, u - T_{\boldsymbol{a}}(\psi)> + \alpha_2 L_2(T_{\boldsymbol{b}}(\omega)) \\ + \frac{\gamma_2}{2}||h-T_{\boldsymbol{b}}(\omega)||^2_{L^2(\Omega)} + <\lambda_2, h - T_{\boldsymbol{b}}(\omega)> \end{aligned} \tag{6.17}$$

where L_1 and L_2 are regularisers representing either Total Variation (where we expect jumps in intensity) or L^2 where we expect smooth edges. Non-negativity is imposed implicitly by the transform and by λ_1 and λ_2 for the image and kernel respectively which force them to be close to their respective non-negative representations.

Minimising with respect to each of the arguments, we have

$$\partial f/\partial u : h^T(hu - z) + \lambda_1 + \gamma_1(u - (B_1\psi + \tilde{R}_1)) = 0, \tag{6.18}$$

$$\partial f/\partial \psi : -B_1\lambda_1 - \gamma_1 B_1\left(u - (B_1\psi + \tilde{R}_1)\right) + L_1(\tilde{\psi})\psi = 0, \tag{6.19}$$

$$\partial f/\partial h : u^T(uh - z) + \lambda_2 + \gamma_2(h - (B_2\omega + \tilde{R}_2)) = 0, \tag{6.20}$$

$$\partial f/\partial \omega : -B_2\lambda_2 - \gamma_2 B_2\left(h - (B_2\omega + \tilde{R}_2)\right) + L_2(\tilde{\omega})\omega = 0 \tag{6.21}$$

where

$$B_1 = \frac{\partial}{\partial\psi} T_{\boldsymbol{a}}(\psi)\Big|_{\psi=0}, \qquad B_2 = \frac{\partial}{\partial\omega} T_{\boldsymbol{b}}(\omega)\Big|_{\omega=0},$$

$$\tilde{R}_1 = R_1\left(\tilde{\psi}\right) = T_{\boldsymbol{a}}(\tilde{\psi}) - B_1\tilde{\psi}, \qquad \tilde{R}_2 = R_2\left(\tilde{\omega}\right) = T_{\boldsymbol{b}}(\tilde{\omega}) - B_2\tilde{\omega},$$

$$L_1\left(\tilde{\psi}\right)\psi = \frac{4a_2 e^{\frac{-2\tilde{\psi}}{a_3}}(a_1+2a_4)\left(a_2 e^{\frac{-2\tilde{\psi}}{a_3}}-1\right)|\nabla\psi|}{\left(1+a_2 e^{\frac{-2\tilde{\psi}}{a_3}}\right)^3 a_3^2} - \nabla\cdot\left(\frac{2(a_1+2a_4)a_2 e^{\frac{-2\tilde{\psi}}{a_3}}}{\left(1+a_2 e^{\frac{-2\tilde{\psi}}{a_3}}\right)^2 a_3|\nabla\tilde{\psi}|}\nabla\psi\right),$$

$$L_2\left(\tilde{\omega}\right)\omega = \frac{4b_2 e^{\frac{-2\tilde{\omega}}{b_3}}(b_1+2b_4)\left(b_2 e^{\frac{-2\tilde{\omega}}{b_3}}-1\right)|\nabla\omega|}{\left(1+b_2 e^{\frac{-2\tilde{\omega}}{b_3}}\right)^3 b_3^2} - \nabla\cdot\left(\frac{2(b_1+2b_4)b_2 e^{\frac{-2\tilde{\omega}}{b_3}}}{\left(1+b_2 e^{\frac{-2\tilde{\omega}}{b_3}}\right)^2 b_3|\nabla\tilde{\omega}|}\nabla\omega\right).$$

We solve equations (6.18) and (6.20) efficiently using Fourier transforms and employ an iterative technique to solve equations (6.19) and (6.21). We present our algorithm in Algorithm 18.

6.4.6 A Mixed Model Suitable for Smooth Blur Kernels

A significant advantage of using the Total Variation (TV) semi-norm in image processing is its ability to assist in the recovery of sharp edges of images and accommodate jumps in intensity. Here we have two quantities to restore. For the image u, the TV is usually satisfactory with the possible issue of staircasing effect for smooth images. For the burring kernel h, the TV is useful when attempting to recover blurring kernels for images blurred by motion or out of focus blur which are piecewise constants (as mentioned in [54]). However when recovering blur functions which are smooth, such as those given by Gaussian functions, a piecewise constant approximation to h can lead to major artefacts in u to the extent of failing the model.

In an attempt to improve the result of recovered smooth (such as Gaussian) kernels, we introduce the following functional which uses the L^2-norm to regularise the blurring kernel $h = T_{\boldsymbol{b}}(\omega)$ and the TV to regularise the image $u = T_{\boldsymbol{a}}(\psi)$, as a hybrid model of (6.2) and (6.3):

$$f_{TM}(\psi,\omega) = \frac{1}{2}||T_{\boldsymbol{b}}(\omega) * T_{\boldsymbol{a}}(\psi) - z||^2_{L^2(\Omega)} + \alpha_1\int_\Omega |\nabla T_{\boldsymbol{a}}(\psi)| + \frac{\alpha_2}{4}\int_\Omega |\nabla T_{\boldsymbol{b}}(\omega)|^2\,\mathrm{d}\Omega. \quad (6.22)$$

We aim to minimise equation (6.22) in order to obtain the Euler-Lagrange equations which we will use to recover the kernel and the image. Since we have discussed the minimisation of the fitting term and the Total Variation regularisation term, it remains to focus on the L^2 term under a transform. This term will be minimised when its partial derivative with respect to ω is equal to zero. Since the partial derivative of $|\nabla h|^2$ with respect to h is given by Δh, it makes sense to calculate this using the chain

Algorithm 18 The Second Constrained Transform Based Algorithm with ADM (TM2)

function $(h, u) \leftarrow$TM2$(z, \alpha_1, \alpha_2, \boldsymbol{a}, \boldsymbol{b}, h^{(0)}, \gamma_1, \gamma_2, tol, maxit)$
  $u^{(0)} \leftarrow z, \;\; \lambda_1^{(0)} \leftarrow \mathbf{0}, \quad \lambda_2^{(0)} \leftarrow \mathbf{0}$
  Calculate $\boldsymbol{a} = \{a_1, a_2, a_3, a_4\}, \quad$ and $\quad \boldsymbol{b} = \{b_1, b_2, b_3, b_4\}$
  $\psi^{(0)} \leftarrow -(a_3/2) \log\left(\left(a_1 + a_4 - u^{(0)}\right) / \left(a_2(u^{(0)} + a_4)\right)\right)$
  $\omega^{(0)} \leftarrow -(b_3/2) \log\left(\left(b_1 + b_4 - h^{(0)}\right) / \left(b_2(h^{(0)} + b_4)\right)\right)$
  for $\ell \leftarrow 1$ **to** $maxit$ **do**
    Solve $\partial f / \partial h = 0$ (equation 6.20) for $h^{(\ell+1)}$, i.e.

$$\left(u^{(\ell)}\right)^T \left(u^{(\ell)} h^{(\ell+1)} - z\right) + \lambda_2^{(\ell)} + \gamma_2 \left(h^{(\ell+1)} - \left(B_2 \omega^{(\ell)} + \tilde{R}_2^{(\ell)}\right)\right) = 0$$

    for $\ell_2 \leftarrow 1$ **to** $maxit$ **do**
      Solve $\partial f / \partial \omega = 0$ (equation 6.21) for $\omega^{(\ell_2+1)}$ given $\omega^{(\ell_2)}$, i.e.

$$-B_2 \lambda_2^{(\ell)} - \gamma_2 B_2 \left(h^{(\ell+1)} - \left(B_2 \omega^{(\ell_2+1)} + R_2^{(\ell_2)}\right)\right) + L_2\left(\omega^{(\ell_2)}\right) \omega^{(\ell_2+1)} = 0$$

    end for
    $\lambda_2^{(\ell+1)} \leftarrow \lambda_2^{(\ell)} + \gamma_2 \left(h^{(\ell+1)} - T_{\boldsymbol{b}}\left(\omega^{(\ell+1)}\right)\right)$
    Solve $\partial f / \partial u = 0$ (equation 6.18) for $u^{(\ell+1)}$, i.e.

$$\left(h^{(\ell+1)}\right)^T \left(h^{(\ell+1)} u^{(\ell+1)} - z\right) + \lambda_1^{(\ell)} + \gamma_1 \left(u^{(\ell+1)} - \left(B_1 \psi^{(\ell)} + \tilde{R}_1^{(\ell)}\right)\right) = 0$$

    for $\ell_1 \leftarrow 1$ **to** $maxit$ **do**
      Solve $\partial f / \partial \psi = 0$ (equation 6.19) for $\psi^{(\ell_1+1)}$ given $\psi^{(\ell_1)}$, i.e.

$$-B_1 \lambda_1^{(\ell)} - \gamma_1 B_1 \left(u^{(\ell+1)} - \left(B_1 \psi^{(\ell_1+1)} + R_1^{(\ell_1)}\right)\right) + L_1\left(\psi^{(\ell_1)}\right) \psi^{(\ell_1+1)} = 0$$

    end for
    $\lambda_1^{(\ell+1)} \leftarrow \lambda_1^{(\ell)} + \gamma_1 \left(u^{(\ell+1)} - T_{\boldsymbol{a}}\left(\psi^{(\ell+1)}\right)\right)$
  end for
  $h \leftarrow h^{(\ell+1)}$
  $u \leftarrow u^{(\ell+1)}$
end function

rule, obtaining[4]

$$\frac{\partial}{\partial\omega}\left|\nabla T_{\boldsymbol{b}}(\omega)\right|^2 = \Delta T_{\boldsymbol{b}}(\omega)\frac{\partial h}{\partial\omega}$$

$$= \frac{2\left(b_1+2b_4\right)b_1E}{\left(1+b_2E\right)^2b_3}\left(\frac{2\left(b_2E-1\right)}{b_3\left(1+b_2E\right)}\nabla\omega\cdot\nabla\omega+\Delta\omega\right)\frac{\partial h}{\partial\omega}$$
$$= \left(\frac{2\left(b_1+2b_4\right)b_1E}{\left(1+b_2E\right)^2b_3}\right)^2\left(\frac{2\left(b_2E-1\right)}{b_3\left(1+b_2E\right)}\nabla\omega\cdot\nabla\omega+\Delta\omega\right)$$

where $A=b_1+2b_4$, $E=E(\omega)=e^{\frac{-2\omega}{b_3}}$ and $B=B(\omega)=1+b_2E$. We may then use the above work to formulate the Euler Lagrange equations for this functional given by (6.23)-(6.24):

$$B_2^2\left(v_2\left(\psi,\tilde{\psi}\right)\right)^T\left(v_2\left(\psi,\tilde{\psi}\right)*\omega-\bar{z}_2\left(\tilde{\psi},\psi,\tilde{\omega}\right)\right)+L_2\left(\tilde{\omega}\right)\omega=0, \tag{6.23}$$

$$B_1^2\left(v_1\left(\omega,\tilde{\omega}\right)\right)^T\left(v_1\left(\omega,\tilde{\omega}\right)*\psi-\bar{z}_1\left(\tilde{\omega},\omega,\tilde{\psi}\right)\right)+L_1\left(\tilde{\psi}\right)\psi=0, \tag{6.24}$$

where

$$\begin{aligned}
v_2\left(\psi,\tilde{\psi}\right) &= B_1\psi+R_1\left(\tilde{\psi}\right),\\
\bar{z}_2\left(\psi,\tilde{\psi}\tilde{\omega}\right) &= B_2^{-1}\bar{z}-B_2^{-1}B_1R_2\left(\tilde{\omega}\right)*\psi,\\
L_2(\tilde{\omega})\omega &= \left(\frac{2\left(b_1+2b_4\right)b_1E}{\left(1+b_2E\right)^2b_3}\right)^2\left(\frac{2\left(b_2E-1\right)}{b_3\left(1+b_2E\right)}\nabla\omega\cdot\nabla\omega+\Delta\omega\right),\\
v_1\left(\omega,\tilde{\omega}\right) &= B_2\omega+R_2\left(\tilde{\omega}\right),\\
\bar{z}_1\left(\omega,\tilde{\omega}\tilde{\psi}\right) &= B_1^{-1}\bar{z}-B_1^{-1}B_2R_1\left(\tilde{\psi}\right)*\omega,\\
L_1(\tilde{\psi})\psi &= \frac{4\tilde{E}_1(a_1+2a_4)(\tilde{E}_1-1)|\nabla\psi|}{(1+\tilde{E}_1)^3a_3^2}-\nabla\cdot\left(\frac{2(a_1+2a_4)\tilde{E}_1}{(1+\tilde{E}_1)^2a_3|\nabla\tilde{\psi}|}\nabla\psi\right).
\end{aligned}$$

Discretisation of the L^2 regularisation term

For the L^2 regularisation term, we implement forward differences to give the approximation to first order derivatives as follows

$$\begin{aligned}
(\nabla\omega\cdot\nabla\omega)_{i,j} &= (\omega_x^2)_{i,j}+(\omega_y^2)_{i,j} = \left(\omega_{i,j}-\omega_{i-1,j}\right)\left(\omega_{i+1,j}-\omega_{i,j}\right)+\left(\omega_{i,j}-\omega_{i,j-1}\right)\left(\omega_{i,j+1}-\omega_{i,j}\right)\\
&= -2\omega_{i,j}^2+\left(\omega_{i+1,j}+\omega_{i-1,j}+\omega_{i,j+1}+\omega_{i,j-1}\right)\omega_{i,j}-\omega_{i-1,j}\omega_{i+1,j}-\omega_{i,j+1}\omega_{i,j-1}
\end{aligned}$$

[4]See Appendix 6.B for details.

Our discretisation is then given by

$$\begin{aligned}\left(L_2\left(T_{\boldsymbol{b}(\omega)}\right)\right)_{i,j} &= \left(A\left(B\nabla\omega\cdot\nabla\omega+\Delta\omega\right)\right)_{i,j}\\ &= A_{i,j}\left(B_{i,j}\left(-2\omega_{i,j}^2+\left(\omega_{i+1,j}+\omega_{i-1,j}+\omega_{i,j+1}+\omega_{i,j-1}\right)\omega_{i,j}-\omega_{i-1,j}\omega_{i+1,j}\right.\right.\\ &\quad \left.\left.-\omega_{i,j+1}\omega_{i,j-1}\right)-4\omega_{i,j}+\omega_{i-1,j}+\omega_{i+1,j}+\omega_{i,j-1}+\omega_{i,j+1}\right)\end{aligned}$$

where

$$A=\left(\frac{2\left(b_1+2b_4\right)b_1E}{\left(1+b_2E\right)^2b_3}\right)^2,\qquad B=\frac{2\left(b_2E-1\right)}{b_3\left(1+b_2E\right)}$$

$$A_{i,j}=\left(\frac{2\left(b_1+2b_4\right)b_1E_{i,j}}{\left(1+b_2E_{i,j}\right)^2b_3}\right)^2,\qquad B_{i,j}=\frac{2\left(b_2E_{i,j}-1\right)}{b_3\left(1+b_2E_{i,j}\right)}.$$

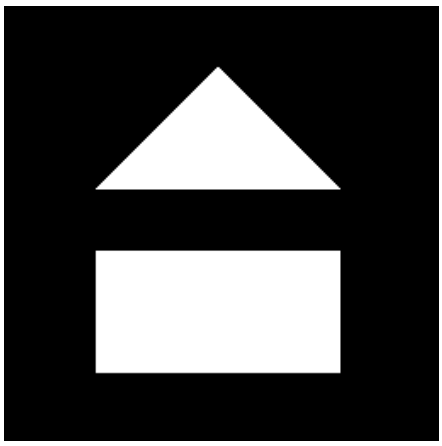
We can now build the coefficient matrix C^{L_2} and residual matrix R^{L_2} such that $L_2(T_{\boldsymbol{b}}(\omega))=C^{L_2}\omega+R^{L_2}$ as

$$\begin{aligned}C^{L_2}_{i+n(j-1),i+n(j-1)} &= -2A_{i,j}\left(B_{i,j}\omega_{i,j}+2\right)\\ \left.\begin{array}{r}C^{L_2}_{i+n(j-1),i+n(j-1)+1}\\ C^{L_2}_{i+n(j-1),i+n(j-1)-1}\\ C^{L_2}_{i+n(j-1),i+nj}\\ C^{L_2}_{i+n(j-1),i+n(j-2)}\end{array}\right\} &= A_{i,j}\left(B\omega_{i,j}+1\right)\\ R^{L_2}_{i+n(j-1),i+n(j-1)} &= -A_{i,j}B_{i,j}\left(\omega_{i-1,j}\omega_{i+1,j}+\omega_{i,j-1}\omega_{i,j+1}\right).\end{aligned}$$

6.5 Experimental Results

The aim of our experimental tests is to demonstrate the effectiveness of our new transform model for restoring both the image u and the kernel h, given the received image z. The results will illustrate the capability of our new algorithm for potentially wide applications. Comparison with previous and competing methods have been shown earlier; here we simply need to show how the new algorithm can better restore these examples and a range of images.

For experimental testing, we consider four key solution sets (see Figure 6.3), including one simple artificial image (Im_1), one real image with a significant amount of dark space (Im_2), one real and more detailed image (Im_3), one real image with fine details (Im_4), one medical image with some fine details (Im_5) and one medical image with many fine details (Im_6). The images have been corrupted by either out of focus blur (Bl_1) or Gaussian blur (Bl_2) (see Figure 6.4). Our aim is to show that the model is able to recover the edges of images as well as many of the details. We aim to recover the finer details in the case of motion blur and fine details in the more challenging case of Gaussian blur. For clarity and consitency, we denote Algorithm 17 by New_6^1 and Algorithm 18 by New_6^2.

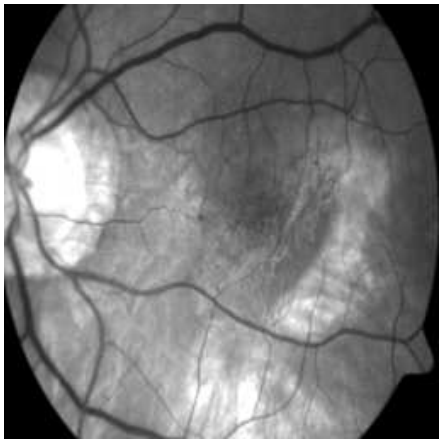


Im_1 - Box-Triangle

Im_2 - Satellite

Im_3 - Barbara

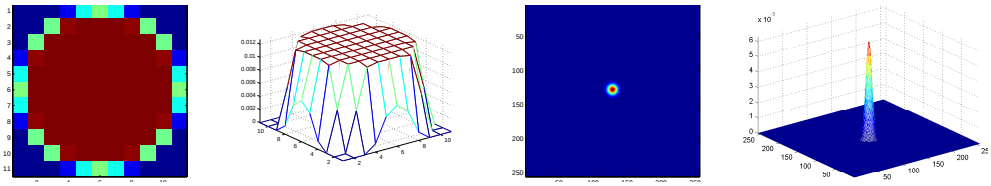
Im_4 - Skyscraper



Im_5 - Retina

Im_6 - Vessels

Figure 6.3: Test case images for experimental results.



Bl_1 - Out of Focus Blur Bl_2 - Gaussian Blur

Figure 6.4: Examples of blur functions used for experimental tests. In each case, we have on image view of the blur function on the left and a mesh view of the same function on the right.

Set 1 — Simple image with blurs

Result set 1 consists of Im_1 corrupted by motion blur (Bl_1) or Gaussian blur (Bl_2). We see in Figure 6.5 that our model is able to reconstruct the edges and preserve the smoothness of the images in the case of motion blur and offers a significant improvement in the case of Gaussian blur.

Set 2 — Detailed image containing many zero-points with blurs

Result set 2 consists of Im_2 corrupted by motion blur (Bl_1) or Gaussian blur (Bl_2). We see in Figure 6.6 that our model is able to preserve the black space well and reconstruct details in the case of motion blur. It is also able to restore some detail in the more challenging case of Gaussian blur.

Set 3 — Detailed photograph images with blurs

Result set 3 consists of Im_3 and Im_4 corrupted by motion blur (Bl_1) or Gaussian blur (Bl_2). We see in Figure 6.7 that in the case of motion blur our model is able to sharpen the images and recover many detailed features including fine details but can introduce pattern defects. In the case of Gaussian blur, we see in Figure 6.8 that many features are recovered in the image and background objects can be distinguished. The intensity ranges are also preserved.

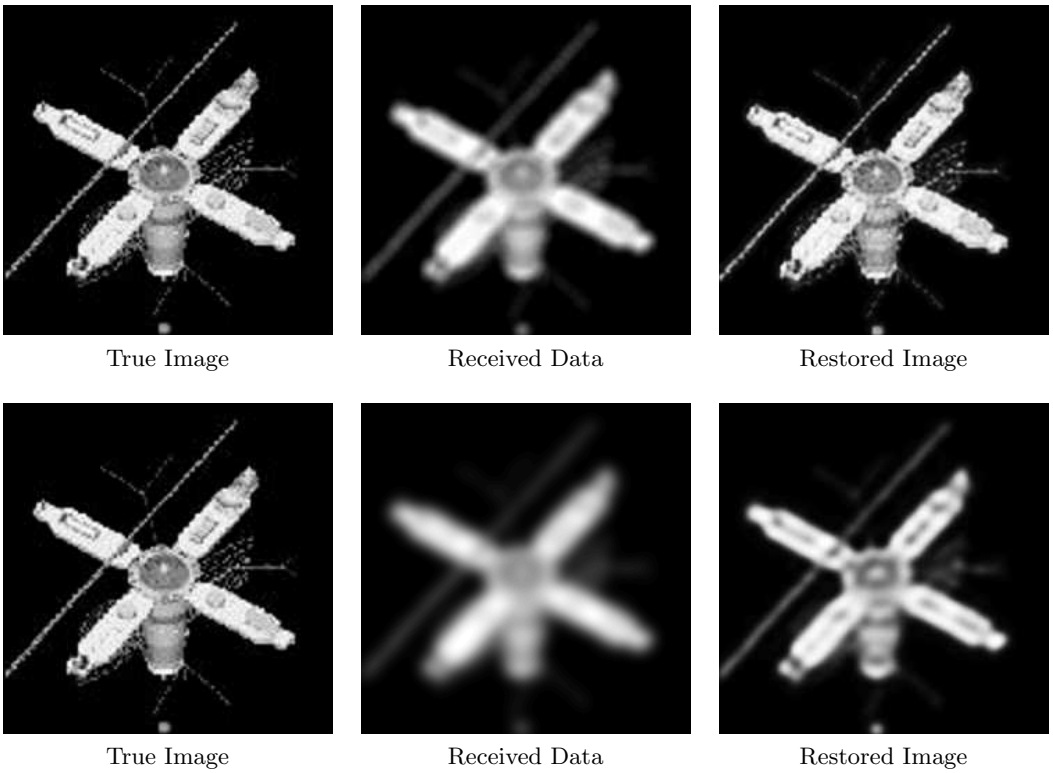
Set 4 — Detailed medical images with blurs

Result set 4 consists of Im_5 and Im_6 corrupted by motion blur (Bl_1) or Gaussian blur (Bl_2). In Figures 6.9 and 6.11, we see that in the case of motion blur our model is able to sharpen the images and recover many fine details. In the case of Gaussian blur, we see in Figure 6.10 that many features are recovered in the image, including blood vessels which were hidden by the blur. The intensity ranges are also preserved.

True Image Received Data Restored Image

True Image Received Data Restored Image

Figure 6.5: Row 1, l-r: Im_1, received data corrupted by Bl_1, restored image using New_6^1. Row 2, l-r: Im_1, received data corrupted by Bl_2, restored image using New_6^1. Our model is capable of restoring edges and preserving black space.



True Image | Received Data | Restored Image

True Image | Received Data | Restored Image

Figure 6.6: Row 1, l-r: Im_2, received data corrupted by Bl_1, restored image using New_6^1. Row 2, l-r: Im_2, received data corrupted by Bl_2, restored image using New_6^1. Our model is capable of restoring details in both cases and of preserving black space.

True Image | Received Data | Restored Image

True Image | Received Data | Restored Image

Figure 6.7: Row 1, l-r: Im_3, received data corrupted by Bl_1, restored image using New_6^1. Row 2, l-r: Im_4, received data corrupted by Bl_1, restored image using New_6^1. Our model is capable of restoring many detailed features and some fine details as well as sharpening edges. There are very few defects in the restored image, notably surrounding the rope in the restored Im_4.

True Image | Received Data | Restored Image

True Image | Received Data | Restored Image

Figure 6.8: Row 1, l-r: Im_3, received data corrupted by Bl_2, restored image using New_6^1. Row 2, l-r: Im_4, received data corrupted by Bl_2, restored image using New_6^1. In the more challenging case of Gaussian blur, our model is capable of restoring some detailed features, including the books in the background of Im_3 and the buildings in Im_4.

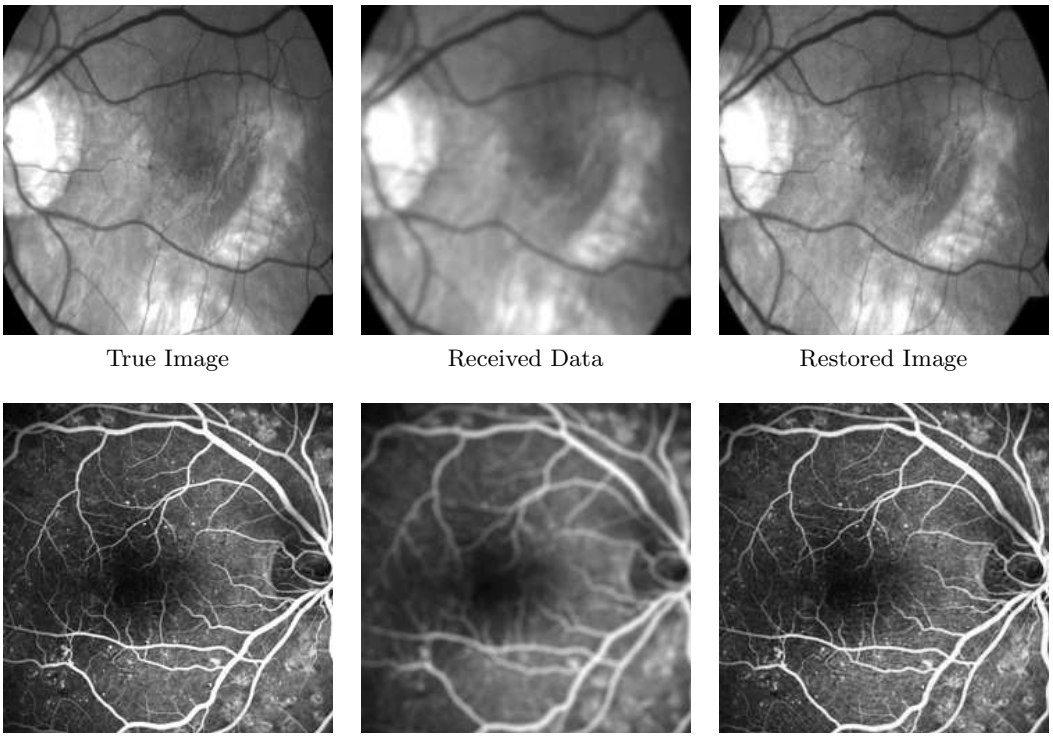
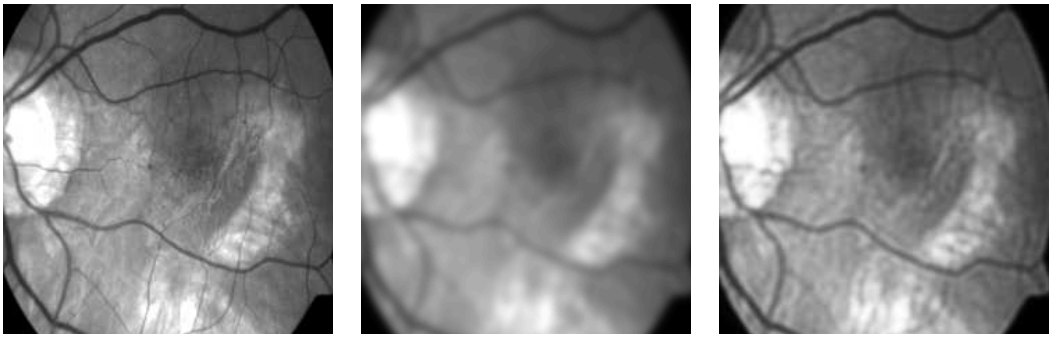
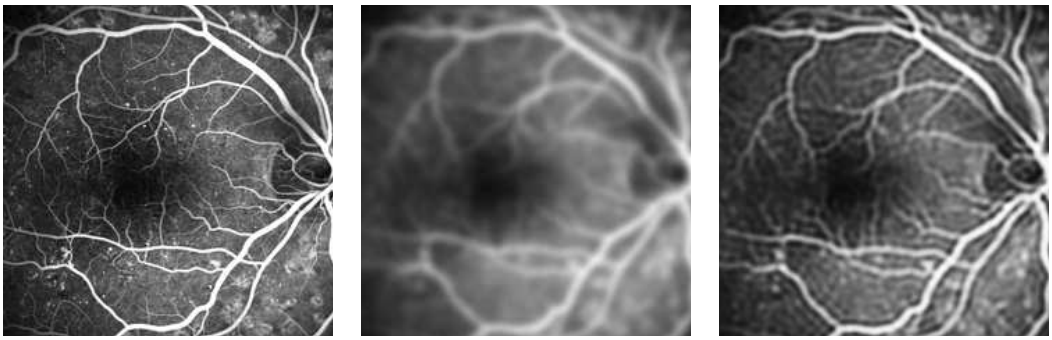


Figure 6.9: Row 1, l-r: Im_5, received data corrupted by Bl_1, restored image using New_6^1. Row 2, l-r: Im_6, received data corrupted by Bl_1, restored image using New_6^1. Our model is capable of restoring many detailed features and sharpen edges. Several of the blood vessels are made visible in Im_5 and some very fine details can be distinguished in Im_6.

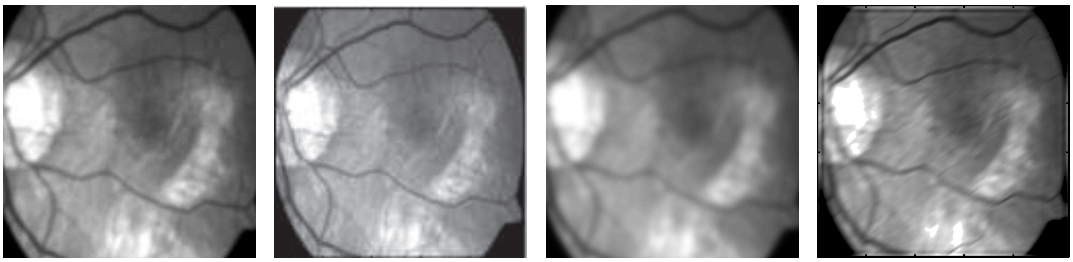


True Image Received Data Restored Image



True Image Received Data Restored Image

Figure 6.10: Row 1, l-r: Im_3, received data corrupted by Bl_2, restored image using New_6^1. Row 2, l-r: Im_4, received data corrupted by Bl_2, restored image using New_6^1. Our model is capable of restoring some detailed features in these challenging cases. Much of the detail is restored in both cases.



(a) Received Data (b) Restored Image (c) Received Data (d) Restored Image

Figure 6.11: (a) Im_3 corrupted by Bl_1, (b) restored image using New_6^2, (c) Im_3 corrupted by Bl_2, (b) restored image using New_6^2. Our accelerated model is capable of obtaining good quality results. Much of the detail is restored in both cases.

6.6 Conclusion

We have presented a total variation based blind deconvolution model with solution positivity achieved by implicit transforms and two solution algorithms for reconstructing a deblurred image along with its blur kernel. We demonstrated that we can ensure positivity and keep the correct range of the image intensities in the case of several blur types, extending the original Chan-Wong model's applicability. This model is particularly effective in reconstructing the kernel without significant defects which can significantly impair the results of previous blind deconvolution algorithms. Our work here can be extended to incorporate the split Bregman method [86] for faster solution and other regularisers beyond the use of first order derivatives [26, 23, 56, 133]. Further work involves integrating the remaining constraints into the functional and automatic selection of regularisation parameters. While there has been work in the parameter selection with non-blind imaging models [93, 95, 218, 59, 205], further work is required to develop a method for the selection of optimal parameters for both regularisation terms in the blind model.

6.A Derivation of Euler Lagrange Equations for the Blind Model (6.8)

Considering the minimisation, we do it in parts. Note that when we minimise with respect to one part, for example when we minimise with respect to the image, the lagged component of the transform is assumed to be equal to the unlagged part, i.e. $\omega = \tilde{\omega}$. We may therefore write for the minimisation with respect to ψ:

$$\begin{aligned}
&\frac{\partial}{\partial\psi}\left(\frac{1}{2}||B_1T_{\boldsymbol{b}}(\omega)*\psi-\bar{z}||^2_{L^2(\Omega)}\right) && (6.25)\\
&= \frac{1}{2}\frac{\partial}{\partial\psi}\left(B_1T_{\boldsymbol{b}}(\omega)*\psi-\bar{z}\right)^2 && (6.26)\\
&= \left(\frac{\partial}{\partial\psi}\left(B_1T_{\boldsymbol{b}}(\omega)*\psi-\bar{z}\right)\right)\left(B_1T_{\boldsymbol{b}}(\omega)*\psi-\bar{z}\right) && (6.27)\\
&= \left(B_1T_{\boldsymbol{b}}(\omega)\right)^T*\left(B_1T_{\boldsymbol{b}}(\omega)*\psi-\bar{z}\right) && (6.28)\\
&= B_1\left(T_{\boldsymbol{b}}(\omega)\right)^T*\left(B_1T_{\boldsymbol{b}}(\omega)*\psi-\bar{z}\right) && (6.29)
\end{aligned}$$

and so we have for the image

$$\frac{\partial}{\partial\psi}\left(\frac{1}{2}||B_1T_{\boldsymbol{b}}(\omega)*\psi-\bar{z}||^2_{L^2(\Omega)}\right) = B_1\left(T_{\boldsymbol{b}}(\omega)\right)^T*\left(B_1T_{\boldsymbol{b}}(\omega)*\psi-\bar{z}\right)$$

and similarly for the kernel

$$\frac{\partial}{\partial\omega}\left(\frac{1}{2}||B_2T_{\boldsymbol{a}}(\psi)*\omega-\bar{z}||^2_{L^2(\Omega)}\right) = B_2\left(T_{\boldsymbol{a}}(\psi)\right)^T*\left(B_2T_{\boldsymbol{a}}(\psi)*\omega-\bar{z}\right).$$

For the second term, see Appendix 5.B for the minimisation of the TV semi-norm

$TV(T_{\boldsymbol{a}}(\psi)) = \int_\Omega |\nabla T_{\boldsymbol{a}}(\psi)|$.

Combining the resulting equation with the fitting term, we obtain the Euler Lagrange equations:

$$B_1(T_{\boldsymbol{b}}(\omega))^T * (B_1 T_{\boldsymbol{b}}(\omega) * \psi - \bar{z}) - \alpha_1 L_1\left(\tilde{\psi}\right) = 0 \tag{6.30}$$

$$B_2(T_{\boldsymbol{a}}(\psi))^T * (B_2 T_{\boldsymbol{a}}(\psi) * \omega - \bar{z}) - \alpha_2 L_2\left(\tilde{\omega}\right) = 0 \tag{6.31}$$

where

$$L_1(\tilde{\psi})\psi = \frac{4\tilde{E}_1(a_1+2a_4)(\tilde{E}_1-1)|\nabla\psi|}{(1+\tilde{E}_1)^3 a_3^2} - \nabla\cdot\left(\frac{2(a_1+2a_4)\tilde{E}_1}{(1+\tilde{E}_1)^2 a_3|\nabla\tilde{\psi}|}\nabla\psi\right)$$

$$L_2(\tilde{\omega})\omega = \frac{4\tilde{E}_1(a_1+2a_4)(\tilde{E}_1-1)|\nabla\omega|}{(1+\tilde{E}_1)^3 a_3^2} - \nabla\cdot\left(\frac{2(a_1+2a_4)\tilde{E}_1}{(1+\tilde{E}_1)^2 a_3|\nabla\tilde{\omega}|}\nabla\omega\right).$$

6.B Derivation of L^2 Regularisation Term for the Mixed Model

We would like to calculate the minimisation of $|\nabla T_{\boldsymbol{b}}(\omega)|^2$ i.e. when its derivative with respect to ω is equal to zero. Since we know that $\frac{\partial}{\partial h}|\nabla h|^2 = \Delta h$, letting $h = T_{\boldsymbol{b}}(\omega)$ we have $\frac{\partial}{\partial\omega}|\nabla T_{\boldsymbol{b}}(\omega)|^2 = \Delta T_{\boldsymbol{b}}(\omega)\frac{\partial h}{\partial\omega}$.

Letting $A = b_1 + 2b_4$, $E = E(\omega) = e^{\frac{-2\omega}{b_3}}$ and $B = B(\omega) = 1 + b_2 E$, we have the derivatives

$$\frac{\partial}{\partial x}E = \frac{\partial}{\partial x}e^{\frac{-2\omega}{b_3}} = e^{\frac{-2\omega}{b_3}}\frac{\partial}{\partial x}\left(\frac{-2\omega}{b_3}\right) = -2Eb_3^{-1}\frac{\partial\omega}{\partial x}$$

and

$$\frac{\partial}{\partial x}B^n = nB^{n-1}\frac{\partial}{\partial x}B = nB^{n-1}\frac{\partial}{\partial x}(b_2E) = nB^{n-1}b_2\frac{\partial}{\partial x}E = -2nB^{n-1}b_2b_3^{-1}E\frac{\partial\omega}{\partial x}.$$

We will also need the second derivatives of the transform with respect to x and y. We give the derivation with respect to x. The derivation with respect to y is similar.

$$\begin{aligned}\frac{\partial}{\partial x}T_{\boldsymbol{b}}(\omega) &= \frac{\partial}{\partial x}AB^{-1} = \left(\frac{\partial}{\partial x}A\right)B^{-1} + A\left(\frac{\partial}{\partial x}B^{-1}\right) = -AB^{-2}\frac{\partial}{\partial x}B \\ &= 2AB^{-2}b_2b_3^{-1}E\frac{\partial\omega}{\partial x}\end{aligned}$$

and so we may calculate the second derivative as follows

$$\begin{aligned}
\frac{\partial^2}{\partial x^2}T_{\boldsymbol{b}}(\omega) &= \frac{\partial}{\partial x}\left(2AB^{-2}b_2b_3^{-1}E\frac{\partial\omega}{\partial x}\right)\\
&= 2Ab_2b_3^{-1}\frac{\partial}{\partial x}\left(B^{-2}E\frac{\partial\omega}{\partial x}\right)\\
&= 2Ab_2b_3^{-1}\left(\frac{\partial}{\partial x}\left(B^{-2}\right)E\frac{\partial\omega}{\partial x}+B^{-2}\frac{\partial E}{\partial x}\frac{\partial\omega}{\partial x}+B^{-2}E\frac{\partial^2\omega}{\partial x^2}\right)\\
&= 2Ab_2b_3^{-1}\left(4b_2b_3^{-1}B^{-3}E\frac{\partial\omega}{\partial x}-2Eb_3^{-1}B^{-2}\left(\frac{\partial\omega}{\partial x}\right)^2+B^{-2}E\frac{\partial^2\omega}{\partial x^2}\right)\\
&= 2Ab_2b_3^{-1}B^{-2}E\left(4b_2b_3^{-1}B^{-1}\frac{\partial\omega}{\partial x}-2b_3^{-1}\left(\frac{\partial\omega}{\partial x}\right)^2+\frac{\partial^2\omega}{\partial x^2}\right).
\end{aligned}$$

We therefore have

$$\Delta T_{\boldsymbol{b}}(\omega) = 2Ab_2b_3^{-1}B^{-2}E\left(4b_2b_3^{-1}B^{-1}\nabla\omega-2b_3^{-1}\nabla\omega\cdot\nabla\omega+\Delta\omega\right)$$

and finally

$$\begin{aligned}
\frac{\partial h}{\partial\omega} &= \frac{\partial}{\partial\omega}T_{\boldsymbol{b}}(\omega)=\frac{\partial}{\partial\omega}\left(AB^{-1}\right)=\left(\frac{\partial}{\partial\omega}A\right)B^{-1}A\frac{\partial}{\partial\omega}B^{-1}\\
&= -AB^{-2}\frac{\partial}{\partial\omega}B=-AB^{-2}b_2\frac{\partial}{\partial\omega}E=2Ab_2b_3^{-1}B^{-2}E.
\end{aligned}$$

We can therefore present the derivation by

$$\begin{aligned}
\frac{\partial}{\partial\omega}\left|\left|\nabla T_{\boldsymbol{b}}(\omega)\right|\right|^2_{L^2(\Omega)} &= \nabla T_{\boldsymbol{b}}(\omega)\frac{\partial h}{\partial\omega}\\
&= 4A^2b_2^2b_3^{-2}B^{-4}E^2\left(4b_2b_3^{-1}B^{-1}\nabla\omega-2b_3^{-1}\nabla\omega\cdot\nabla\omega+\Delta\omega\right).
\end{aligned}$$

Chapter 7

Semi-Blind Deblurring with Parametric Kernel Identification

In this chapter, we aim to achieve similar deblurring results to those presented in Chapter 6, making use of assumptions or available information in order to reduce the time required to give a solution. We achieve this by considering the case of semi-blind deblurring of images. We present in §7.3 some enhancements to existing work which offer improved results over current models. We construct new formulations which are better able to cope with additive noise in blurred images, making some standard assumptions about the blur function which may be estimated from visual observation. A method for implicitly imposing constraints on image intensity values, introduced in Chapter 5, is used which has been found to provide improved results over competing methods. We demonstrate that this enhanced model offers improved results for Gaussian type blur, particularly in the presence of noise. Then in §7.4, we present formulations to model alternative causes of blur degradation as differentiable functions which may be used in a parametric deblurring framework and show that this allows us to obtain improved results over alternative parametric methods.

7.1 Introduction

Parametric kernel identification can be used to deblur images making some assumptions about the blur function. There exist many models which work well without noise. However, noise is often present in images and can cause misidentification of the blur function. This paper presents a model for restoring noisy, blurred images in which the blur is assumed to be of a certain type.

Semi-blind deblurring [2, 11] is one of the three main types of deconvolution problem. It involves recovering the hidden true image with only partial knowledge or assumption about the blur function, such as the type of blur [123, 32]. Such models perform well and can obtain improved results over blind deblurring when the blur type may be known or estimated. Such techniques are useful in related areas such as the segmentation of blurred images [11] (see Chapter 8) and super-resolution [130, 217]

where the blur is often of Gaussian or out of focus type. Parametric deblurring is a type of semi-blind deblurring which assumes that we may be able to model the blur degradation term h as a parametric function dependent on only a few parameters. Our aim in this case is to recover the parameters and thus reconstruct the blur function.

The presence of noise in images, as discussed in Chapter 3, can be a particular problem in all types of deblurring, which is typically addressed by using regularisation [169, 149, 153, 152, 21] which aims to offset the ill-posedness of the problem. This task brings with it the issue of selecting regularisation parameters [21, 59, 70, 93, 95, 3] which remains a problem, particularly in blind deblurring. We present a parametric model which uses alternate direction methods [197, 71] and regularisation to separate the problem of denoising from identifying the true image and blur function. We also incorporate implicit constraints introduced in Chapter 5. This model offers improved results over competing methods and appears to be robust.

Much research involving parametric models either assumes that Gaussian blur is the cause for degradation or models alternative blur functions as piecewise constant functions. Assuming that the blur is of Gaussian type is a limitation which may prevent the accurate restoration of images corrupted by other blur types, so we would like to extend this to accommodate other blurs. We note that modelling a blur function as a piecewise constant prohibits the recovery of the parameters within a variational framework. We therefore present differentiable equations which closely approximate some common alternative types of blur, such as out of focus, and demonstrate their effectiveness in recovering the image.

This chapter is organised as follows. In §7.2, we present existing ideas for dealing with parametric kernel identification and deblurring. In §7.3, we introduce the new models for dealing with blurred and noisy images. In §7.4, we present alternative formulations for dealing with other blur types focusing on out of focus blur, box blur and linear motion blur. In §7.5, we present experimental results. In §7.6, we present conclusions of this work.

7.2 Existing Models

As discussed in §3.3.8, many papers attempt to improve the estimation of point spread functions by considering them as piecewise functions after discretisation. This reduces the estimation of many unknowns to that of only a few variables which determine the entire point spread function (psf).

In [18, 78, 117, 9], the authors give several equations for piecewise constant representations of kernel functions, including motion blur, out of focus blur and box blur. The task is to estimate the correct parameters to form the blur function before using the resulting psf to deblur the image using filtering techniques or variational modelling.

Since Gaussian blur has a well-known differentiable formulation given by

$$h^{\mathcal{G}}(x, y, \sigma) \;=\; \frac{1}{2\pi\sigma^2} \exp\left(-\frac{x^2+y^2}{2\sigma^2}\right), \tag{7.1}$$

which is dependent on the variable σ, recent papers aimed at semi-blind deconvolution assuming Gaussian blur such as [11, 64] have introduced this formulation into the functional. . In [11], the objective functional for the restoration is similar to total variation regularised deblurring [169, 54] made up of a fitting term for deblurring in the presence of Gaussian noise and a smoothness term given as the L^2-norm of the gradient which allows for smooth kernels. The equation is modified to allow for the parametrically defined kernel function h_σ. It is thus presented as

$$\mathcal{F}_{BSK}(u, \sigma) = \frac{1}{2}\int_\Omega (h^{\mathcal{G}} * u - z)^2 dA + \gamma \int_\Omega |\nabla h_\sigma|^2 dA$$

where γ is a regularisation parameter controlling the smoothness of the recovered blur function, u is the restored image and z is the received data. The authors then minimise this functional alternately with respect to the image u and the argument σ of the blur function.

7.3 A New Model for Implicitly Constrained Semi-Blind Deconvolution

In this section, we introduce three enhancements of model (7.2) which aim to improve the restored image by constraining the intensity range of the image implicitly (7.6) and attempting to offset the ill-effects of noise (7.10) leading to our main model (7.11) which is able to successfully recover images from noisy and blurred data.

In parametric deblurring, we can often avoid regularisation and attempt to identify the blur function using only a fitting term, aiming to minimise this energy with some regularisation for the image in the semi-blind deblurring case. We solve the problem

$$\min_{\sigma,u}\left\{F(\sigma, u) = \frac{1}{2}||h^{\mathcal{G}}(x, y, \sigma) * u - z||^2_{L^2(\Omega)} + \alpha_u R_u(u)\right\} \tag{7.2}$$

where R_u is a regularisation term for the image and α_u is a small positive parameter. There are several ideas which are aimed at selecting the best parameter α_u or at least automatically selecting a value which gives a good restoration of the image, traditionally in the non-blind case but also more recently in the blind case [59, 70, 3, 93, 95]. We also note that there are many ideas for alternative regularisation in the literature (see for example [26, 23, 56, 133]) which can typically improve the restoration but this is not our primary concern. We use the total variation (tv) regularisation given by

$$R_u(u) = \int_\Omega |\nabla u|$$

due to it's ability to recover edges of an image and solving the first order optimality conditions obtained from the derivatives of F with respect to u and σ:

$$\frac{\partial F}{\partial \sigma} = \int_{\Omega} \left(\frac{\partial h^{\mathcal{G}}(x,y,\sigma)}{\sigma} * u \right) \left(h^{\mathcal{G}}(x,y,\sigma) * u - z \right) \mathrm{d}\Omega, \tag{7.3}$$

$$\frac{\partial F}{\partial u} = \bar{h}^{\mathcal{G}}(x,y,\sigma) \left(h^{\mathcal{G}}(x,y,\sigma) * u - z \right) + \alpha_u \frac{\partial R_u(u)}{\partial u}. \tag{7.4}$$

where $\bar{h}^{\mathcal{G}}(x,y,\sigma) = h^{\mathcal{G}}(-x,-y,\sigma)$ and the derivative of the regularisation term with respect to u is calculated as

$$\frac{\partial R_u(u)}{\partial u} = \nabla \cdot \left(\frac{\nabla u}{|\nabla u|_\beta} \right), \ |\nabla u|_\beta = \sqrt{u_x^2 + u_y^2 + \beta^2}$$

where β is a small positive parameter which is included to avoid division by zero. One attractive reason for modelling the blur parametrically is that we have an analytic equation for the blur function and so may calculate the derivatives rather than approximate them.

We attempt to recover the blur function and the image simultaneously using alternate minimisation, as in [54] and, using the estimate of the blur function at each iteration, recover the image using preconditioned conjugate gradient (pcg). The main attraction of parametric modelling of the kernel is that we may use a fast solver in order to recover the parameter σ. We may use Newton's method or even the bisection method in order to give the solution assuming that the image is known.

7.3.1 Enhancement 1: Incorporating Implicit Constraints

It is often the case, particularly in astronomical imaging and blind deblurring that restored intensity values fall outside of the range of the true image which successively deteriorates the quality of the restored image and kernel. As discussed in Chapter 5, we have found that by using an appropriate transform in place of the image given by $\mathcal{T} = \mathcal{T}_{\boldsymbol{a}}(\psi) : \mathbb{R} \to \mathbb{R}$ for some function $\psi : \Omega \subset \mathbb{R}^2 \to \mathbb{R}$ defined on the domain Ω such the upper and lower limits of $\mathcal{T}$ are constrained, we can better control control the range of intensity values in the recovered image. The idea here is to recover ψ and, with an appropriate formulation, attempt to force u to be close to $\mathcal{T}$. Such suitable formulation is provided by the alternate direction method (ADM) [44, 104, 198, 188, 197]. One suitable transform function may be given by

$$\mathcal{T}_{\boldsymbol{a}}(\psi) = \frac{a_1 + 2a_4}{1 + a_2 e^{-2\frac{\psi}{a_3}}} - a_4 \tag{7.5}$$

with suitably chosen $\boldsymbol{a}$. Substituting this function into Equation (7.2) and using ADM, we derive our new functional (7.6):

$$\max_{\lambda_\psi} \min_{\sigma,u,\psi} \left\{ F_1(\sigma,u,\psi) = \frac{1}{2} ||h^{\mathcal{G}}(\sigma) * u - z||^2_{L^2(\Omega)} + F_u(u,\psi,\lambda_\psi) \right\} \tag{7.6}$$

where

$$F_u(u,\psi,\lambda_\psi) \;=\; \alpha_\psi R_\psi(\mathcal{T}_{\boldsymbol{a}}(\psi)) + \alpha_u R_u(u) \\ + \gamma_\psi ||\mathcal{T}_{\boldsymbol{a}}(\psi) - u||^2_{L^2(\Omega)} + < \lambda_\psi, \mathcal{T}_{\boldsymbol{a}}(\psi) - u >, \tag{7.7}$$

α_ψ, γ_ψ are small positive parameters, $\lambda_\psi \in \mathbb{R}^2$ and R_ψ denotes non-linear regularisation for the transformed function. Note that, since we effectively have non-linear regularisation for the image given by the regularisation of the transformed ψ, we may relax the regularisation for the image u in order to improve the speed of solution by using a linear regulariser such as Tikhonov [193].

The Euler-Lagrange equation for σ is given by Equation (7.3) and the Euler-Lagrange equation for u is given by

$$\frac{\partial F}{\partial u} = \bar{h}^{\mathcal{G}}(x,y,\sigma)\left(h^{\mathcal{G}}(x,y,\sigma) * u - z\right) + \alpha_u \frac{\partial R_u(u)}{\partial u} + \gamma_\psi(u - \mathcal{T}_{\boldsymbol{a}}(\psi)) + \lambda_\psi(u - \mathcal{T}_{\boldsymbol{a}}(\psi)) \tag{7.8}$$

with the derivative of the regulariser given by $\partial R_u(u)/\partial u = u$ or $\partial R_u(u)/\partial u = \Delta u$ for Tikhonov or L^2 respectively and so (7.8) can now be solved efficiently using Fourier transforms. It remains to give the first order optimality condition for ψ:

$$\frac{\partial F_1}{\partial \psi} \;=\; \alpha_\psi \frac{R_\psi(\mathcal{T}_{\boldsymbol{a}}(\psi))}{\partial \psi} + \gamma_\psi \frac{\partial \mathcal{T}_{\boldsymbol{a}}(\psi)}{\partial \psi}(\mathcal{T}_{\boldsymbol{a}}(\psi) - u) + \lambda_\psi \frac{\partial \mathcal{T}_{\boldsymbol{a}}(\psi)}{\partial \psi}(\mathcal{T}_{\boldsymbol{a}}(\psi) - u). \tag{7.9}$$

Since this equation is non-linear for ψ, we use time marching to solve within the alternate minimisation framework. The overall algorithm is presented in Algorithm 19.

Algorithm 19 Constrained Parametric Deblurring (CPD1)

function $(\sigma, u) \leftarrow$CPD1$(\sigma^{(0)}, \alpha_u, \alpha_\psi, \gamma_\psi, maxit, tol)$
  Initial Estimates
  $u^0 \leftarrow z$
  $\psi^{(0)} \leftarrow \xi_{\boldsymbol{a}}\left(u^0\right)$
  $\lambda_\psi^0 \leftarrow \mathbf{1}$
  for $\ell \leftarrow 1$ **to** $maxit$ **do**
    Solve for the image using alternate direction method:
    for $\ell_2 \leftarrow 1$ **to** $maxit$ **do**
      Solve for u using (7.8) and Fourier transforms or pcg
      Solve for ψ using (7.9) and time marching
      Update λ_ψ:

$$\lambda_\psi^{(\ell)} \leftarrow \lambda_\psi^{(\ell-1)} + \gamma_\psi\left(\tau_{\boldsymbol{a}}\left(\psi^{(\ell)}\right) - u^{(\ell)}\right)$$

    end for
    Solve for kernel parameter σ using (7.3) and bisection
  end for
end function

7.3.2 Enhancement 2: Regularisation of the Blur Function

In this section we introduce a model which aims at reducing defects caused by noise in the received data by introducing regularisation for the blur function into the model given by equation (7.6), providing the new problem

$$\max_{\lambda_\psi} \min_{\sigma,u,\psi} \{ F_2(\sigma, u, \psi, \lambda_\psi) = \frac{1}{2}||h^{\mathcal{G}}(\sigma) * u - z||^2_{L^2(\Omega)} + \alpha_\sigma R_\sigma(\sigma) + F_u(u, \psi, \lambda_\psi)\} \quad (7.10)$$

where for the regularisation term R_σ, we use either L^2 regularisation or total variation regularisation of the blurring function given by

$$R_\sigma(\sigma) = ||\nabla h(\sigma)||^2_{L^2(\Omega)} \quad \text{and} \quad R_\sigma(\sigma) = \int_\Omega |\nabla h(\sigma)|$$

respectively and whose derivatives are given earlier. As demonstrated in Figure 7.8, this allows for correction of the perturbation of the parameter σ but does not appear sufficiently reliable and is dependent heavily on the choice of parameter α_σ.

7.3.3 Enhancement 3: Smoothing Noise

We now aim to improve our approximation of σ by separating the noise from the blurred and noisy image z. We replace this term in the functional with a new term v which should be the denoised z, such that $v \approx h^{\mathcal{G}}(\sigma) * u$. We use alternate direction minimisation in order to attempt to drive the terms z and v closer together in a manner similar to denoising.

We present this functional as

$$\max_{\lambda_\psi,\lambda_z} \min_{\sigma,u,\psi,v} \{F_3(\sigma, u, \psi, \lambda_\psi, v, \lambda_z) = \frac{1}{2}||h^{\mathcal{G}}(\sigma) * u - v||^2_{L^2(\Omega)}$$

$$+ F_u(u, \psi, \lambda_\psi) + \alpha_\sigma R_\sigma(\sigma) + \frac{\gamma_z}{2}||v - z||_{L^2(\Omega)} + < \lambda_z, v - z > \Big\}, \quad (7.11)$$

where F_u is given by Equation (7.7) and we use a linear regularisation method for the image. This yields the optimality conditions given by

$$\frac{\partial F_3}{\partial \sigma} = \int_\Omega \left(\frac{\partial h^{\mathcal{G}}(\sigma)}{\partial \sigma} * u \right) \left(h^{\mathcal{G}}(\sigma) * u - v \right) \mathrm{d}\Omega + \alpha_\sigma \frac{R_\sigma(\sigma)}{\partial \sigma} \quad (7.12)$$

$$\frac{\partial F_3}{\partial u} = \bar{h}^{\mathcal{G}}(\sigma) \left(h^{\mathcal{G}}(\sigma) * u - v \right) + \alpha_u \frac{\partial R_u(u)}{\partial u} \quad (7.13)$$

$$\frac{\partial F_3}{\partial v} = v - h^{\mathcal{G}}(\sigma) * u + \gamma_z(v - z) + \lambda_z(v - z) \quad (7.14)$$

$$\frac{\partial F_3}{\partial \psi} = \alpha_\psi \frac{R_\psi(\mathcal{T}_{\boldsymbol{a}}(\psi))}{\partial \psi} + \gamma_\psi \frac{\partial \mathcal{T}_{\boldsymbol{a}}(\psi)}{\partial \psi}(\mathcal{T}_{\boldsymbol{a}}(\psi) - u) + \lambda_\psi \frac{\partial \mathcal{T}_{\boldsymbol{a}}(\psi)}{\partial \psi}(\mathcal{T}_{\boldsymbol{a}}(\psi) - u). \quad (7.15)$$

We solve model (7.11) using alternate minimisation of the arguments, letting the initial estimates for u and v be given by the received data z and letting the initial estimate for ψ be given by the inverse transform of the initial estimate for u. Initial

estimates may also be given for σ from visual observation, however a default value of 1 may be used, accompanied by an initial step-size. We solve (7.12) using bisection since it is able to recover the parameter quickly, (7.13) may be solved using Fourier transforms, (7.14) may solved directly and (7.15) is solved using time marching. We present the overall algorithm for this model in Algorithm 20.

Algorithm 20 Constrained Parametric Deblurring (CPD2)

function $(\sigma, u) \leftarrow$CPD2$(\sigma^{(0)}, \alpha_u, \alpha_\psi, \alpha_\sigma, \gamma_\psi, \gamma_z, maxit, tol)$
  Initial Estimates
  $u^0 \leftarrow z$
  $v^0 \leftarrow z$
  $\psi^{(0)} \leftarrow \xi_{\boldsymbol{a}}\left(u^0\right)$
  $\lambda_\psi^0 \leftarrow \mathbf{1}$
  $\lambda_z^0 \leftarrow \mathbf{1}$
  for $\ell \leftarrow 1$ **to** $maxit$ **do**
    Solve (7.14) for v using time marching
    Solve (7.13) for u using Fourier transforms or pcg
    Solve (7.15) for ψ using time marching
    Update λ_ψ and λ_z:

$$\begin{cases} \lambda_\psi^{(\ell)} \leftarrow \lambda_\psi^{(\ell-1)} + \gamma_\psi\left(\tau_{\boldsymbol{a}}\left(\psi^{(\ell)}\right) - u^{(\ell)}\right) \\ \lambda_z^{(\ell)} \leftarrow \lambda_z^{(\ell-1)} + \gamma_z\left(v^{(\ell)} - z\right) \end{cases}$$

    Solve (7.12) for the kernel parameter σ using bisection
  end for
end function

7.4 Constructing Alternative Blur Functions

Much work in parametric kernel identification assumes that the blur function is given by a Gaussian kernel. While this is a common blur type, it does not allow for recovery from alternative blur types, such as linear motion. We present in this section an equation which may replace the Gaussian function $h^{\mathcal{G}}$ to allow for such reconstructions. Beginning with a well-known analytical approximation to the 1-dimensional Heaviside step function, which tends to the Heaviside function $H(x)$ as $\varepsilon \to 0$, we extend this to 2 dimensions

$$h(x, y, \sigma) = 1 - \frac{1}{1 + \exp\left(-2\frac{(x-x_c)^2 + (y-y_c)^2}{\chi}\right)} \tag{7.16}$$

and add structure and some constraints which aim to keep the continuous functions close to the piecewise constant representation of the point spread function at the discretisation step to build our approximations, given by Equations (7.43) - (7.44) for three alternative blur types which, after discretisation should be close to the true representations of the blur.

We will use level sets

$$\mathcal{L}_c(f) = \{(x, y) | f(x, y) = c\}$$

to describe the shape which we aim to achieve. We call $\mathcal{L}_c(f)$ the *c-level set of* f.

7.4.1 Out of Focus Blur

The out of focus blur (or averaging filter) function has the effect of averaging the pixel intensity values in a local (assumed circular) neighbourhood. As such, this degradation can be modelled as $z = h * u$ where u is the true image, z is the blurred image, and h is a piecewise constant function which is equal to $\omega(\sigma)$ inside a circle of radius σ and zero elsewhere, i.e.

$$h^{\mathcal{OD}}(x, y, \sigma) = \begin{cases} \omega(\sigma) & \text{if } x^2 + y^2 \leq \sigma \\ 0 & \text{otherwise.} \end{cases} \tag{7.17}$$

We would like to model this as a differentiable function. We begin with the function (7.16) and attempt to force it to approximate (7.17) by adding a parameter ϕ and a function $\omega(\sigma)$ depending only on σ, obtaining

$$h^{\mathcal{O}}(x, y, \sigma) = \omega(\sigma) - \frac{\omega(\sigma)}{1 + \phi \exp\left(-2\frac{(x-x_c)^2+(y-y_c)^2}{\chi}\right)}. \tag{7.18}$$

We would like to impose the following constraints:

1. When the c-level set of the function defines a circle of radius σ, c should be close to the maximal function value $\omega(\sigma)$:

   $$(x - x_c)^2 + (y - y_c)^2 = \sigma^2 \quad \Leftrightarrow \quad h^{\mathcal{O}}(x, y, \sigma) = (1 - \delta_1)\,\omega(\sigma)$$

   for some small δ_1, say $\delta_1 = 10^{-2}, 10^{-3}$. From this, we have

   $$\phi = \phi_{\delta_1,\chi}(\sigma) = \frac{1 - \delta_1}{\delta_1} \exp\left(\frac{\sigma^2}{\chi}\right). \tag{7.19}$$

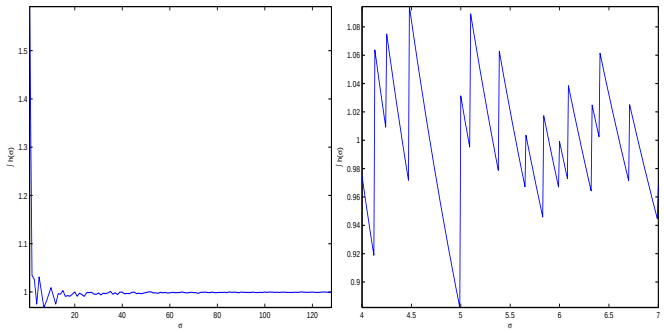
   Substituting (7.19) into (7.18) gives us

   $$h^{\mathcal{O}}(x, y, \sigma) = \omega(\sigma) - \frac{\omega(\sigma)}{1 + \frac{1-\delta_1}{\delta_1} \exp\left(-\frac{(x-x_c)^2+(y-y_c)^2-\sigma^2}{\chi}\right)} \tag{7.20}$$

2. When the c-level set of the function defines a circle of radius slightly larger than σ, c should be close to zero:

   $$(x - x_c)^2 + (y - y_c)^2 = (\sigma + \delta_2)^2 \quad \Rightarrow \quad h^{\mathcal{O}}(x, y, \sigma) = \delta_1 \omega(\sigma)$$

   for some small $\delta_2 \leq 1/n$, say $\delta_2 = 10^{-2}, 10^{-3}$. From this and equation (7.20), we



$f(\sigma)$, $\sigma \in (1, n/2)$ $f(\sigma)$, $\sigma \in (4, 7)$

Figure 7.1: Graphs of the integral $f(\sigma) = \int_\Omega h^{\mathcal{O}}(x, y, \sigma)\,\mathrm{d}\Omega$ for $n = 256$ and varying σ. Note that while the integral appears to tend toward the unit for larger values of σ, it is rarely equal to the unit for lower (more realistic) values.

have

$$\chi = \chi_{\delta_1,\delta_2}(\sigma) = \frac{-\delta_2\,(2\sigma + \delta_2)}{2\ln\left(\frac{\delta_1}{1-\delta_1}\right)}. \tag{7.21}$$

Substituting (7.21) into (7.20) gives us

$$h^{\mathcal{O}}(x, y, \sigma) = \omega(\sigma) - \frac{\omega(\sigma)}{1 + \left(\frac{\delta_1}{1-\delta_1}\right)^{2\frac{(x-x_c)^2+(y-y_c)^2-\sigma^2}{\delta_2(2\sigma+\delta_2)}-1}} \tag{7.22}$$

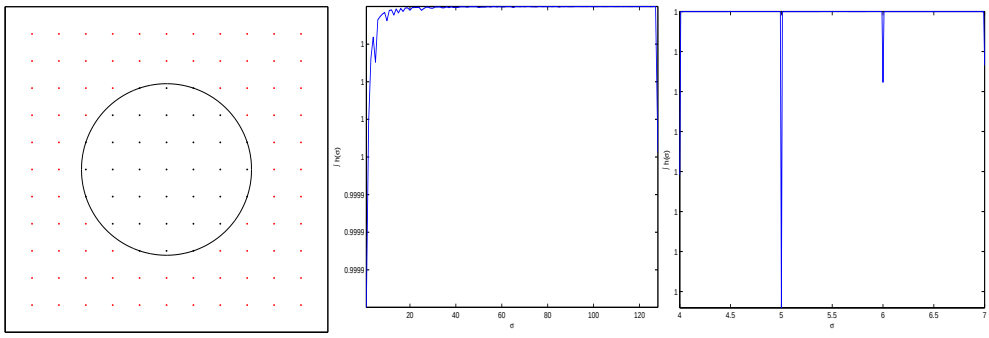
It remains to give the function ω, the value of which gives the maximum value of the function h. Since the shape of this function resemble a cylinder of radius σ and volume equal to one (since we require a unit integral), it is tempting to write $\omega = \omega(\sigma) = 1/\pi\sigma^2$. While this is commonly used and may be considered accurate, this function ω may not achieve a unit integral of h over the domain Ω (see Figure 7.1).

This causes the average pixel intensity value of the convolution $h * u$ to be unstable for varying σ after discretisation and creates many local minima of $||h^{\mathcal{O}}(x, y, \sigma) * u(x, y) - z(x, y)||$with respect to σ (see Figure 7.3). As a solution to this, we might consider the function $w(\sigma) = 1/N(\sigma)$ where

$$N(\sigma) = \#\left\{(x, y)|x^2 + y^2 \leq \sigma, (x, y) \in \mathbb{Z} \times \mathbb{Z}\right\}$$

is the number of lattice points lying within a circle of radius σ (including the boundary), but this is not differentiable. We therefore let the function $\omega(\sigma) = 1$ and normalise the values of the kernel function, giving

$$h^{\mathcal{O}}(x, y, \sigma) = \frac{\bar{h}^{\mathcal{O}}(x, y, \sigma)}{\int_\Omega \bar{h}^{\mathcal{O}}(x, y, \sigma)\,\mathrm{d}\Omega} \tag{7.23}$$



(a) Circle of radius $\sqrt{10}$ on a lattice

(b) $f(\sigma)$ for $\sigma \in (1, n/2)$

(c) $f(\sigma)$ for $\sigma \in (4, 7)$

Figure 7.2: Illustration of the Gauss Circle problem and the ability to preserve a unit integral using it: a) Circle of radius $\sigma = \sqrt{10}$ on a lattice. The red points are outside of the circle while the black points, of which there are $N(\sigma) = 37$, are inside the circle or on the boundary, b) Graph of the integral $f(\sigma) = \int_\Omega h(x, y, \sigma)\,\mathrm{d}\Omega$, c) zoomed in graph of the integral.

where the equation $\bar{h}^{\mathcal{O}}$ is given by

$$\bar{h}^{\mathcal{O}}(x, y, \sigma) = 1 - \frac{1}{1 + \left(\frac{\delta_1}{1-\delta_1}\right)^{2\frac{(x-x_c)^2+(y-y_c)^2-\sigma^2}{\delta_2(2\sigma+\delta_2)}-1}}. \tag{7.24}$$

7.4.2 Box Blur

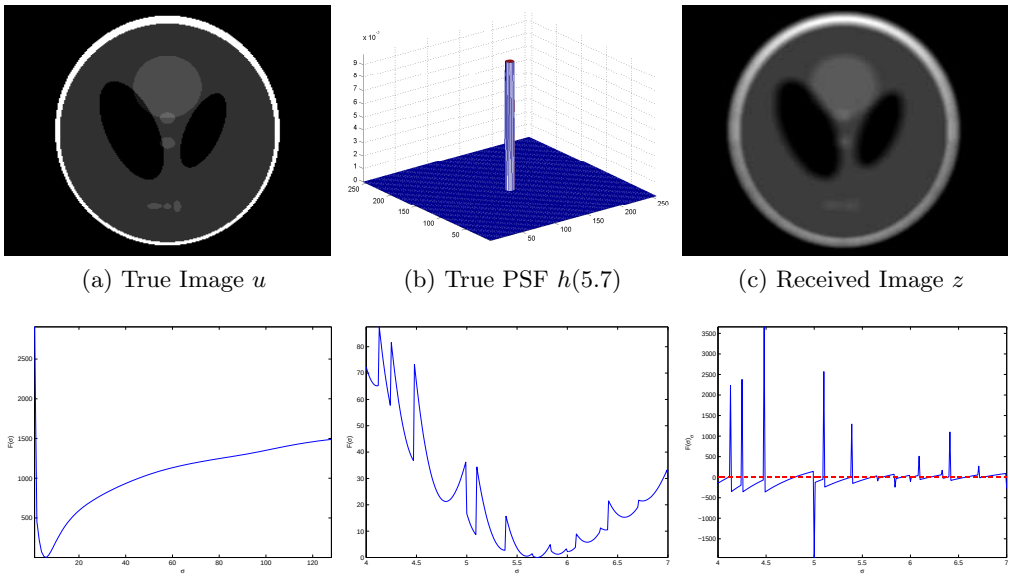
In order to construct a function to give box blur, we begin with equation (7.18) and attempt to expand this to a square by increasing the power. We present the initial box blur function as

$$h^{\mathcal{B}}(x, y, \sigma) = w(\sigma) - \frac{w(\sigma)}{1 + \phi \exp\left(-\frac{(x-x_c)^p+(y-y_c)^p}{\chi}\right)} \tag{7.25}$$

where σ gives the radius of the blur function, and x_c and y_c are the x and y centre-points respectively.

We now want to impose similar constraints to those we impose for out of focus blur such that the intensity will be close to the upper limit of the blur function on the $2\sigma \times 2\sigma$ square and the that the value drops sharply to almost zero when σ is increased by a small amount. We also want to impose a constraint which means that the blur function should be as box-like as necessary and give a formula for calculating the power p. Assuming that we know we have box-blur, we do not need to consider p as a variable. We state our condition as

1. When the c-level set of the function defines a square of width 2σ, c should be



(a) True Image u (b) True PSF $h(5.7)$ (c) Received Image z

(d) $F(\sigma)$, $\sigma \in (1, n/2)$ (e) $F(\sigma)$, $\sigma \in (4, 7)$ (f) $R(\sigma) = F_\sigma$, $\sigma \in (4, 7)$

Figure 7.3: Illustration of the many local minima of the functional $F(\sigma) = ||h(\sigma) * u - z||_2^2$ against σ for the Phantom image example with an out of focus blur of radius 5.7: a) true image u, b) true kernel $h(x, y, 5.7)$, c) received data $z(x, y) = h(x, y, 5.7) * u(x, y)$, d) $F(\sigma)$ appears to be convex however e) if we look closely near the global minimum there are many local minima resulting in f) many solutions to the minimisation problem.

close to the maximal value $\omega(\sigma)$:

$$(x - x_c)^p + (y - y_c)^p = \sigma^p \quad \Leftrightarrow \quad h^{\mathcal{B}}(x, y, \sigma) = (1 - \delta_1)\,\omega(\sigma).$$

Implementing this condition in equation (7.25), we have

$$(1 - \delta_1)\,\omega(\sigma) = \omega(\sigma) - \frac{\omega(\sigma)}{1 + \phi \exp\left(-\frac{\sigma^p}{\chi}\right)} \quad \Leftrightarrow \quad 1 - \delta_1 = 1 - \frac{1}{1 + \phi \exp\left(-\frac{\sigma^p}{\chi}\right)}$$

$$\Leftrightarrow \quad \delta_1 + \delta_1 \phi \exp\left(-\frac{\sigma^p}{\chi}\right) = 1$$

which gives the equation for ϕ as

$$\phi = \phi_{\delta_1, p, \chi}(\sigma) = \frac{1 - \delta_1}{\delta_1} \exp\left(\frac{\sigma^p}{\chi}\right). \tag{7.26}$$

Substituting equation (7.26) into equation (7.25), we have

$$h^{\mathcal{B}}(x, y, \sigma) = \omega(\sigma) - \frac{\omega(\sigma)}{1 + \frac{1-\delta_1}{\delta_1} \exp\left(-\frac{(x-x_c)^p + (y-y_c)^p - \sigma^p}{\chi}\right)}. \tag{7.27}$$

2. When the c-level set of the function defines a square of width slightly larger than

2σ, c should be close to zero:

$$(x-x_c)^p+(y-y_c)^p=(\sigma+\delta_2)^p \quad \Leftrightarrow \quad h^{\mathcal{B}}(x,y,\sigma)=\delta_1\omega(\sigma)$$

We first note that this condition can be written as

$$(x-x_c)^p+(y-y_c)^p-\sigma^p=\mathcal{A} \quad \Leftrightarrow \quad h^{\mathcal{B}}(x,y,\sigma)=\delta_1\omega(\sigma) \tag{7.28}$$

where $\mathcal{A}=\sum_{k=1}^{p}\frac{p!}{k!(p-k)!}\sigma^{p-k}\delta_2^k$.

Implementing this condition in equation (7.27) gives

$$\delta_1\omega(\sigma)=\omega(\sigma)-\frac{\omega(\sigma)}{1+\frac{1-\delta_1}{\delta_1}\exp\left(-\frac{\mathcal{A}}{p_2}\right)}$$

$$\Leftrightarrow \quad \delta_1=1-\frac{1}{1+\frac{1-\delta_1}{\delta_1}\exp\left(-\frac{\mathcal{A}}{\chi}\right)} \quad \Leftrightarrow \quad 1+\frac{1-\delta_1}{\delta_1}e^{-\frac{\mathcal{A}}{p_2}}=\frac{1}{1-\delta_1}$$

$$\Leftrightarrow \quad \exp\left(-\frac{\mathcal{A}}{\chi}\right)=\left(\frac{\delta_1}{1-\delta_1}\right)^2 \quad \Leftrightarrow \quad -\frac{\mathcal{A}}{\chi}=2\ln\frac{\delta_1}{1-\delta_1}$$

giving us the equation for χ as

$$\chi=\frac{-\mathcal{A}}{2\ln\frac{\delta_1}{1-\delta_1}}. \tag{7.29}$$

Substituting equation (7.29) into equation (7.27), we have

$$\begin{aligned} h^{\mathcal{B}}(x,y,\sigma)&=\omega(\sigma)-\frac{\omega(\sigma)}{1+\frac{1-\delta_1}{\delta_1}\exp\left(2\ln\frac{\delta_1}{1-\delta_1}\frac{(x-x_c)^p+(y-y_c)^p-\sigma^p}{\mathcal{A}}\right)} \\ &=\omega(\sigma)-\frac{\omega(\sigma)}{1+\frac{1-\delta_1}{\delta_1}\left(\exp\left(\ln\frac{\delta_1}{1-\delta_1}\right)\right)^{2\frac{(x-x_c)^p+(y-y_c)^p-\sigma^p}{\mathcal{A}}}}. \end{aligned} \tag{7.30}$$

So we have

$$h^{\mathcal{B}}(x,y,\sigma)=\omega(\sigma)-\frac{\omega(\sigma)}{1+\left(\frac{\delta_1}{1-\delta_1}\right)^{2\frac{(x-x_c)^p+(y-y_c)^p-\sigma^p}{\mathcal{A}}-1}}. \tag{7.31}$$

3. The graph of the $(1-\delta_1)\omega(\sigma)$-level set is close to a square in shape:

$$\sigma-\delta_3+x_c\le x<\sigma+x_c \Leftrightarrow x=y;\ x,y>0 \text{ when } (x-x_c)^p+(y-y_c)^p=\sigma^p$$

where $\delta_3\le 1/n$. We also assume that $x_c=y_c$ which is a reasonable assumption.

To implement this condition, first note that when $x=y$ and $x_c=y_c$, we have

$$(x-x_c)^p+(y-y_c)^p=\sigma^p \quad \Leftrightarrow \quad 2\,(x-x_c)^p=\sigma^p$$

$$\Leftrightarrow \quad 2\sigma^p \left(\frac{x-x_c}{\sigma}\right)^p = \sigma^p \quad \Leftrightarrow \quad \left(\frac{x-x_c}{\sigma}\right)^p = \frac{1}{2} \quad \Leftrightarrow \quad p = \log_{\frac{x-x_c}{\sigma}} \left(\frac{1}{2}\right)$$

From the condition, we have

$$\sigma - \delta_3 + x_c \leq x < \sigma + x_c \quad \Leftrightarrow \quad \log_{\frac{\sigma-\delta_3}{\sigma}} \left(\frac{1}{2}\right) \leq \log_{\frac{x-x_c}{\sigma}} \left(\frac{1}{2}\right) < \underbrace{\log_1 \left(\frac{1}{2}\right)}_{=\infty}$$

which holds if the bases are less than 1. This is the case with these conditions. Therefore, in order to satisfy this condition, we must have

$$\log_{1-\frac{\delta_3}{\sigma}} \left(\frac{1}{2}\right) \leq p < \infty. \tag{7.32}$$

As a final step, we must note that the power p must be an even integer. We therefore set $p = 2 + p_1 - \lfloor p_1 \rfloor_2$ where $p_1 = \log_{1-\delta_3/\sigma} \frac{1}{2}$ and the notation $\lfloor p_1 \rfloor_2$ means ($p_1 \mod 2$). We also consider the unit integral $\int_\Omega h^{\mathcal{B}}(x, y, \sigma)\, \mathrm{d}\Omega = 1$. To enforce this in the discrete case, we note that the volume will be approximately equal to $4\sigma^2\omega(\sigma)$. Since we require the unit integral, we set $w(\sigma) = 1/(4\sigma^2)$ and present our equation for box blur as

$$h^{\mathcal{B}}(x, y, \sigma) = \frac{1}{4\sigma^2} \left(1 - \frac{1}{1 + \left(\frac{\delta_1}{1-\delta_1}\right)^{2\frac{(x-x_c)^p + (y-y_c)^p - \sigma^p}{\mathcal{A}} - 1}}\right) \tag{7.33}$$

where

$$p = 2 + p_1 - \lfloor p_1 \rfloor_2, \qquad p_1 = \log_{1-\delta_3 - \frac{x_c}{r}} \frac{1}{2}, \qquad \mathcal{A} = \sum_{k=1}^{p} \frac{p!}{k!(p-k)!} \sigma^{p-k} \delta_2^k,$$

and δ_1 is a small positive parameter, say 10^{-3} and $0 < \delta_2, \delta_3 \leq 1/n$.

Finally, noting that we will need to find the derivative of $h^{\mathcal{B}}$ with respect to σ, our given equations for p are not ideal. Our solution is to either make p a continuous function which is dependent on σ (which is not possible since we require σ to be a multiple of 2) or set p at the start to satisfy our requirements for any radius. Given (7.32), we have

$$p \geq \max_{\sigma_i \in R} \left\{ \log_{1-\frac{\delta_3}{\sigma_i}} \left(\frac{1}{2}\right) \right\}, \qquad R = \left[1, \ldots, \left\lfloor \frac{n}{2} \right\rfloor\right]. \tag{7.34}$$

Note that $0 \leq \frac{\delta_3}{\sigma_i} < 1 \, \forall \sigma_i \in R$ and importantly $\log_A \left(\frac{1}{2}\right) > \log_B \left(\frac{1}{2}\right)$ if $0 < B < A < 1$, so $p \geq \log_{1-\frac{\delta_3}{\lfloor \frac{n}{2} \rfloor}} \left(\frac{1}{2}\right)$ is equivalent to (7.34). Now, we just need p to be a multiple of 2 so we set $p = 2 + p_1 - \lfloor p_1 \rfloor_2$ where $p_1 = \log_{1-\frac{\delta_3}{\lfloor \frac{n}{2} \rfloor}} \left(\frac{1}{2}\right)$.

We present our final equation as

$$h^{\mathcal{B}}(x, y, \sigma) = \omega(\sigma) - \frac{\omega(\sigma)}{1 + \left(\frac{\delta_1}{1-\delta_1}\right)^{2\frac{(x-x_c)^p + (y-y_c)^p - \sigma^p}{\mathcal{A}} - 1}} \tag{7.35}$$

where

$$\omega(\sigma) = \frac{1}{4\sigma^2}, \qquad p = 2 + p_1 - \lfloor p_1 \rfloor_2 \,,$$

$$p_1 = \log_{1-\frac{\delta_3}{\lfloor \frac{n}{2} \rfloor}} \left(\frac{1}{2}\right), \qquad \mathcal{A} = \mathcal{A}(\sigma) = \sum_{k=1}^{p} \frac{p!}{k!(p-k)!} \sigma^{p-k} \delta_2^k,$$

and δ_1 is a small positive parameter, say 10^{-3} and $0 < \delta_2, \delta_3 \leq 1/n$.

7.4.3 Linear Motion Blur

In this case, we begin with the base equation for out of focus blur given by equation (7.18) and attempt to compress this to an ellipse which has a very small diameter and a perpendicular diameter equal to σ by adding a small non-negative coefficient δ_3 as a divisor of $x - x_c$ and dividing $y - y_c$ by the radius σ which is equal to half the length of the blur.

Instead of a circle, we use the equation of an ellipse given by

$$\left(\frac{x - x_c}{a}\right)^2 + \left(\frac{y - y_c}{b}\right)^2 = 1 \tag{7.36}$$

such that

$$x = x_c \Leftrightarrow y = y_c \pm b, \qquad y = y_c \Leftrightarrow x = x_c \pm a,$$

which we intend to flatten to approximate linear motion blur. We achieve this by letting $a = \delta_3$ for some small δ_3 and $b = \sigma$.

We also wish to be able to rotate this shape. We accomplish this by introducing a variable θ to give the angle of rotation and replace the coordinates (x, y) with

$$(\bar{x}, \bar{y}) = (\bar{x}(\theta), \bar{y}(\theta)) = (x\cos\theta - y\sin\theta, x\sin\theta + y\cos\theta).$$

We present our initial equation for Linear Motion Blur as

$$h^{\mathcal{L}}(x, y, \sigma, \theta) = \omega(\sigma) - \frac{\omega(\sigma)}{1 + \phi \exp\left(-\frac{\left(\frac{\bar{x} - \bar{x}_c}{\delta_3}\right)^2 + \left(\frac{\bar{y} - \bar{y}_c}{\sigma}\right)^2}{\chi}\right)} \tag{7.37}$$

where $\delta_3 \leq 1/n$. We impose the following constraints:

1. When the c-level set defines an ellipse of the form (7.36), c is close to the maximal value $\omega(\sigma)$:

$$\left(\frac{\bar{x} - \bar{x}_c}{\delta_3}\right)^2 + \left(\frac{\bar{y} - \bar{y}_c}{\sigma}\right)^2 = 1 \quad \Leftrightarrow \quad h^{\mathcal{L}}(x, y, \sigma, \theta) = (1 - \delta_1)\omega(\sigma).$$

Applying this condition to equation (7.37), we have

$$(1 - \delta_1)\,\omega(\sigma) = \omega(\sigma) - \frac{\omega(\sigma)}{1 + \phi \exp\left(-\frac{1}{\chi}\right)} \quad \Leftrightarrow \quad \delta_1 = \frac{1}{1 + \phi \exp\left(-\frac{1}{\chi}\right)}$$

$$\Leftrightarrow \quad \delta_1 + \delta_1\phi \exp\left(-\frac{1}{\chi}\right) = 1$$

that is, letting $\epsilon_1 = \delta_1/(1-\delta_1)$

$$\phi = \epsilon_1^{-1} \exp\left(\frac{1}{\chi}\right). \tag{7.38}$$

Substituting (7.38) into equation (7.37), we have

$$h^{\mathcal{L}}(x, y, \sigma, \phi) = \omega(\sigma) - \frac{\omega(\sigma)}{1 + \epsilon_1^{-1} \exp\left(-\frac{\left(\frac{\bar{x}-\bar{x}_c}{\delta_3}\right)^2 + \left(\frac{\bar{y}-\bar{y}_c}{\sigma}\right)^2 - 1}{\chi}\right)}. \tag{7.39}$$

2. When the c-level set defines an ellipse just larger than that defined above, c is close to zero:

$$\left(\frac{\bar{x}-\bar{x}_c}{\delta_3}\right)^2 + \left(\frac{\bar{y}-\bar{y}_c}{\sigma}\right)^2 = \left(1+\frac{\delta_2}{\sigma}\right)^2 \quad \Leftrightarrow \quad h^{\mathcal{L}}(x, y, \sigma, \theta) = \delta_1\omega(\sigma)$$

Applying this condition to equation (7.39), we have

$$\delta_1\omega(\sigma) = \omega(\sigma) - \frac{\omega(\sigma)}{1 + \epsilon_1^{-1} \exp\left(-\frac{\left(1+\frac{\delta_2}{\sigma}\right)^2 - 1}{\chi}\right)}$$

$$\Leftrightarrow \quad (1-\delta_1)\left(1 + \epsilon_1^{-1} \exp\left(-\frac{\left(1+\frac{\delta_2}{\sigma}\right)^2 - 1}{\chi}\right)\right) = 1$$

$$\Leftrightarrow \quad (1-\delta_1)\,\epsilon_1^{-1} \exp\left(-\frac{\left(1+\frac{\delta_2}{\sigma}\right)^2 - 1}{\chi}\right) = \delta_1 \quad \Leftrightarrow \quad -\frac{\left(1+\frac{\delta_2}{\sigma}\right)^2 - 1}{\chi} = 2\ln\epsilon_1$$

that is

$$\chi = \chi_{\delta_1,\delta_2}(\sigma) = -\frac{\left(1+\frac{\delta_2}{\sigma}\right)^2 - 1}{2\ln\epsilon_1}. \tag{7.40}$$

Substituting (7.40) into equation (7.39), we have

$$h^{\mathcal{L}}(x, y, \sigma, \theta) = \omega(\sigma) - \frac{\omega(\sigma)}{1 + \epsilon_1^{-1} \exp\left(2\ln\epsilon_1 \frac{\left(\frac{\bar{x}-\bar{x}_c}{\delta_3}\right)^2 + \left(\frac{\bar{y}-\bar{y}_c}{\sigma}\right)^2 - 1}{\left(1+\frac{\delta_2}{\sigma}\right)^2 - 1}\right)},$$

that is

$$h^{\mathcal{L}}(x, y, \sigma, \theta) = \omega(\sigma) - \frac{\omega(\sigma)}{1 + \epsilon_1^{2\frac{\left(\frac{\bar{x}-\bar{x}_c}{\delta_3}\right)^2 + \left(\frac{\bar{y}-\bar{y}_c}{\sigma}\right)^2 - 1}{\left(1+\frac{\delta_2}{\sigma}\right)^2 - 1} - 1}}. \tag{7.41}$$

Finally, we give the relationship between $w(\sigma)$ and σ as $2\sigma\omega(\sigma) = 1$, therefore

rewriting our function as

$$h^{\mathcal{L}}(x,y,\sigma,\theta) = \frac{1}{2\sigma}\left(1 - \left(1+\epsilon_1^{\,2\frac{\left(\frac{\bar{x}-\bar{x}_c}{\delta_3}\right)^2+\left(\frac{\bar{y}-\bar{y}_c}{\sigma}\right)^2-1}{\left(1+\frac{\delta_2}{\sigma}\right)^2-1}-1}\right)^{-1}\right). \tag{7.42}$$

7.4.4 Combined Equation

In this section, we present a single equation to give the above 3 blur types depending on parameters. Combining the above formulations into one equation allows us to give the derivative only once permitting some brevity.

$$h^{\mathcal{H}_1}(x,y,\sigma,\theta) = \frac{h^{\mathcal{H}_2}(x,y,\sigma,\theta)}{\int_\Omega h^{\mathcal{H}_2}(x,y,\sigma,\theta)\,\mathrm{d}\Omega} \tag{7.43}$$

where we use the integral over Ω to attempt to keep a unit integral for $h^{\mathcal{H}_1}$ over the domain, as in the derivation of the formulation for out of focus blur. We present the equation for $h^{\mathcal{H}_2}$ in (7.43) as

$$h^{\mathcal{H}_2}(x,y,\sigma,\theta) = 1 - \left(1+\varepsilon^{2\frac{\left(\frac{\bar{x}}{c_x}\right)^{P_X}+\left(\frac{\bar{y}}{c_y}\right)^{P_X}-\sigma^{P_\sigma}}{\mathcal{A}}-1}\right)^{-1} \tag{7.44}$$

where $\varepsilon = \frac{\delta_1}{1-\delta_1}$ and δ_1 is a small positive parameter. With suitable adjustment of the parameters, we can achieve the following blur types (see Figure 7.4).

1. Out of focus can be achieved by setting $P_X = P_\sigma = 2$, $\mathcal{A} = \delta_2\left(2\sigma+\delta_2\right)$, $c_x = c_y = 1$, $(\bar{x},\bar{y}) = (x,y)$.

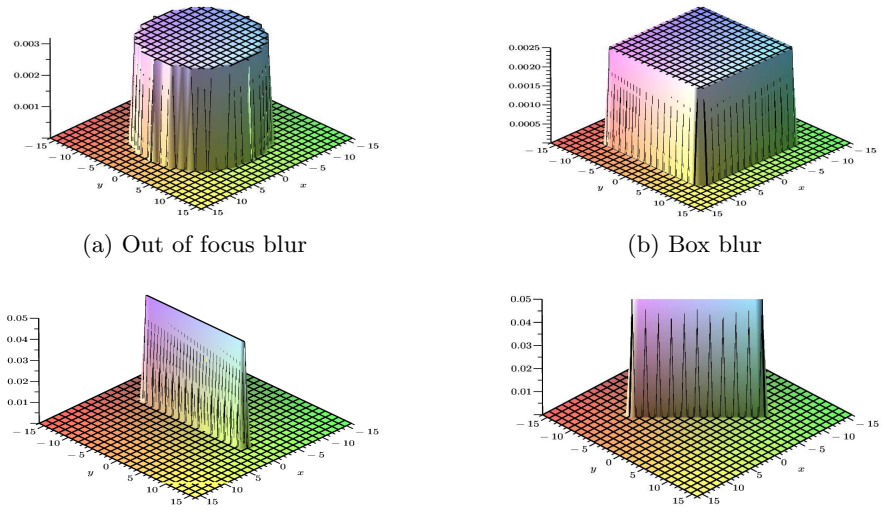
2. Box blur can be achieved by setting $c_x = c_y = 1$, $(\bar{x},\bar{y}) = (x,y)$, $P_X = P_\sigma = 2+p-\lfloor p\rfloor_2$,

$$p = \log_{1-\frac{\delta_3}{\lfloor\frac{n}{2}\rfloor}}\left(\frac{1}{2}\right),\ \mathcal{A} = \sum_{k=1}^{P_X}\frac{P_X!}{k!(P_X-k)!}\sigma^{P_X-k}\delta_2^k.$$

3. Linear motion blur can be achieved by setting $P_X = 2$, $P_\sigma = 0$, $c_x = \delta_3$, $c_y = \sigma$,

$$\mathcal{A} = \left(1+\frac{\delta_2}{\sigma}\right)^2 - 1,\ \begin{pmatrix}\bar{x}\\ \bar{y}\end{pmatrix} = \begin{pmatrix}x\cos\theta - y\sin\theta\\ x\sin\theta + y\cos\theta\end{pmatrix}$$

The parameters $0 < \delta_2, \delta_3 \leq 1/n$ are small and positive. In the cases of out of focus blur and box blur, we aim to minimise with respect to the argument σ which controls the radius and length of the blurs respectively. For the third case, we may minimise with respect to σ to recover the length of the blur or with respect to θ to recover the orientation.



(a) Out of focus blur (b) Box blur

(c) Linear motion blur 1 (d) Linear Motion blur 2

Figure 7.4: Illustration of Parametric Blur functions. We have a) Out of Focus blur $h^{\mathcal{O}}(x, y, \sigma)$ with $\sigma = 10$, b) Box Blur $h^{\mathcal{B}}(x, y, \sigma)$ with $\sigma = 10$, c) Linear Motion Blur $h^{\mathcal{L}}(x, y, \sigma, \theta)$ with $\sigma = 10, \theta = 0$, d) Linear Motion Blur $h^{\mathcal{L}}(x, y, \sigma, \theta)$ with $\sigma = 10, \theta = 3\pi/4$.

We now aim to solve models which make use of this combined function. For this, we require the derivative of the kernel function:

$$\frac{\partial h^{\mathcal{H}_1}(x, y, \sigma, \theta)}{\partial \sigma} = \frac{-h^{\mathcal{H}_2}(x, y, \sigma, \theta) \int_\Omega \frac{\partial h^{\mathcal{H}_2}(x,y,\sigma,\theta)}{\partial \sigma}}{\left(\int_\Omega h^{\mathcal{H}_2}(x, y, \sigma, \theta)\, \mathrm{d}\Omega\right)^2} + \frac{\frac{\partial h^{\mathcal{H}_2}(x,y,\sigma,\theta)}{\partial \sigma}}{\int_\Omega h^{\mathcal{H}_2}(x, y, \sigma, \theta)\, \mathrm{d}\Omega} \tag{7.45}$$

$$\frac{\partial h^{\mathcal{H}_1}(x, y, \sigma, \theta)}{\partial \theta} = \frac{-h^{\mathcal{H}_2}(x, y, \sigma, \theta) \int_\Omega \frac{\partial h^{\mathcal{H}_2}(x,y,\sigma,\theta)}{\partial \theta}}{\left(\int_\Omega h^{\mathcal{H}_2}(x, y, \sigma, \theta)\, \mathrm{d}\Omega\right)^2} + \frac{\frac{\partial h^{\mathcal{H}_2}(x,y,\sigma,\theta)}{\partial \theta}}{\int_\Omega h^{\mathcal{H}_2}(x, y, \sigma, \theta)\, \mathrm{d}\Omega} \tag{7.46}$$

where the derivatives of the sub-function $h^{\mathcal{H}_2}(x, y, \sigma, \theta)$ are given by

$$\frac{\partial h^{\mathcal{H}_2}(x, y, \sigma, \theta)}{\partial \sigma} \tag{7.47}$$

$$= \frac{-2}{V_1^2} \varepsilon^{V_2} \ln(\varepsilon) \left(\frac{\frac{\partial \mathcal{A}}{\partial \sigma}}{\mathcal{A}^2} \left(\left(\frac{\bar{x}}{c_x}\right)^{P_X} + \left(\frac{\bar{y}}{c_y}\right)^{P_X} - \sigma^{P_\sigma} \right) + \frac{1}{\mathcal{A}} \left(\frac{\bar{y}^{P_X} \frac{\partial c_y}{\partial \sigma}}{c_y^{P_X+1}} + P_\sigma \sigma^{P_\sigma - 1} \right) \right) \tag{7.48}$$

$$\frac{\partial h^{\mathcal{H}_2}(x, y, \sigma, \theta)}{\partial \theta} = \frac{2}{V_1^2 \mathcal{A}} \varepsilon^{V_2} \ln(\varepsilon) \left(\frac{P_X \bar{x}^{P_X - 1}}{c_x P_X} \frac{\partial \bar{x}}{\partial \theta} + \frac{P_X \bar{y}^{P_X - 1}}{c_y P_X} \frac{\partial \bar{y}}{\partial \theta} \right) \tag{7.49}$$

where the sub-equations are given by

$$V_1 = 1 + \varepsilon^{V_2}, \qquad V_2 = 2 \frac{\left(\frac{\bar{x}}{c_x}\right)^{P_X} + \left(\frac{\bar{y}}{c_y}\right)^{P_X} - \sigma^{P_\sigma}}{\mathcal{A}} - 1$$

Then for each of the blur types which we are attempting to construct, we must calculate the following:

1. For out of focus blur, we have $\partial\mathcal{A}/\partial\sigma = 2\delta_2$ and $\partial c_y/\partial\sigma = \partial\bar{x}/\partial\theta = \partial\bar{y}/\partial\theta = 0$.
2. For box blur, we have $\partial\mathcal{A}/\partial\sigma = p(\sigma+\delta_2)^{p-1} - p\sigma^{p-1}$ and $\partial c_y/\partial\sigma = \partial\bar{x}/\partial\theta = \partial\bar{y}/\partial\theta = 0$.
3. For linear motion blur, we have $\partial\mathcal{A}/\partial\sigma = -2\delta_2(1+\delta_2/\sigma)/\sigma^2$, $\partial c_y/\partial\sigma = 1$, $\partial\bar{x}/\partial\theta = -\bar{y}$ and $\partial\bar{y}/\partial\theta = \bar{x}$.

We then aim to solve a related minimisation problem to that presented in §7.3.3 and adapted to include the composite blur function as

$$\max_{\lambda_\psi,\lambda_z}\min_{\sigma,\theta,u,\psi,v}\{F_3(\sigma,\theta,u,\psi,\lambda_\psi,v,\lambda_z) = \frac{1}{2}||h^{\mathcal{H}_1}(\sigma,\theta)*u-v||^2_{L^2(\Omega)}$$

$$+ F_u(u,\psi,\lambda_\psi) + \alpha_\sigma R_\sigma(\sigma) + \frac{\gamma_z}{2}||v-z||_{L^2(\Omega)} + <\lambda_z, v-z>\Big\}, \tag{7.50}$$

where F_u is given by Equation (7.7) and we use a linear regularisation method for the image. We then proceed with alternate minimisation of the arguments in a similar manner to that presented in Algorithm 20.

7.5 Experimental Results

In this section we describe the examples to be tested, define our measures of error and present results. We aim to show that while existing models can be effective for images with little or no noise, the new models are also capable in this situation and moreover. We also aim to show that the new models enhance the quality of the result.

7.5.1 Images

For experimental testing we use a combination of artificial images, photographs and medical images: Im_1: Box-Triangle, Im_2: Satellite, Im_3: "Lunch Atop a Skyscraper," Im_4: Barbara, Im_5: Colour Fundus Image, Im_6: Fundus Angiography Retinal Image.

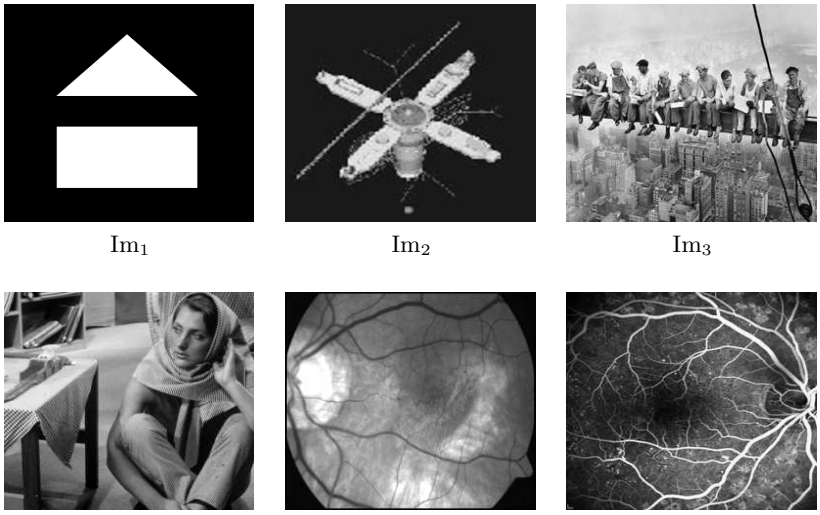
7.5.2 Blur functions

For experimental testing we use Bl_1: Gaussian blur $h^{\mathcal{G}}(x,y,1.2)$, Bl_2: Out of focus blur $h^{\mathcal{O}}(x,y,9)$, Bl_3: Box blur $h^{\mathcal{B}}(x,y,13)$, Bl_4: Linear motion blur $h^{\mathcal{L}}(x,y,13,\pi/2)$.

7.5.3 Models

We test the examples with the following models to compare quality visually and numerically:

- $\textbf{Mod}_1$: The filtering model given in [18].



Im_1 Im_2 Im_3

Im_4 Im_5 Im_6

Figure 7.5: Images used for experimental testing.

- **Mod**$_2$: The filtering model Mod_1, constrained by scaling to keep the correct intensity range of the restored results.
- **Mod**$_3$: The filtering model Mod_1, constrained by truncation to keep the correct intensity range of the restored results.
- **Mod**$_4$: The tv model given by Equation (7.2).
- **Mod**$_5$: The tv model Mod_4, constrained by scaling to keep the correct intensity range of the restored results.
- **Mod**$_6$: The tv model Mod_4, constrained by truncation to keep the correct intensity range of the restored results.
- **New**$_7^1$: The implicitly constrained model given by Equation (7.6).
- **New**$_7^2$: The implicitly constrained regularised model given by Equation (7.10).
- **New**$_7^3$: The implicitly constrained noise separation model given by Equation (7.11).
- **New**$_7^4$: The implicitly constrained noise separation model New_7^3 but assuming the out of focus blur model in place of Gaussian.
- **New**$_7^5$: The implicitly constrained noise separation model New_7^3 but assuming the box blur model.
- **New**$_7^6$: The implicitly constrained noise separation model New_7^3 but assuming the linear motion blur model.

7.5.4 Measuring Error

In order to measure the error in our restored data, we make use of the popular and frequently used measure peak signal to noise ratio (psnr) [36] of the restored data u compared with the true data u_{true}, as introduced in §4.3.1, which tends to infinity as the restored data tends towards the true data. This measure is given by

$$psnr(u,v) = 20\log_{10}\frac{\max(u_{i,j})}{\frac{1}{mn}\sum_{i,j}|u_{i,j}-v_{i,j}|^2}$$

It is considered that the above measure (as well as related measures signal to noise ratio (snr), mean squared error among others) does not give a particularly informative measure for image processing, since they consider only point-wise difference in intensity values. The authors of [199, 200, 201] consider more appropriate error measures using components which compare contrast, luminance and structural similarity of the true and restored images. We give an error measure obtained from this as

$$ssim_{\alpha,\beta,\gamma,C_1,C_2,C_3}(u,v) = \left(\frac{2\mu_u\mu_v + C_1}{\mu_u^2+\mu_v^2+C_1}\right)^{\alpha}\left(\frac{2\rho_u\rho_v + C_1}{\rho_u^2+\rho_v^2+C_2}\right)^{\beta}\left(\frac{\rho_{uv}+C_3}{\rho_u\rho_v+C_3}\right)^{\gamma}$$

where $C_1 = (\max(u)\epsilon_1)^2$, $C_2 = (\max(u)\epsilon_2)^2$ and $C_3 = (\max(u)\epsilon_3)^2$ are present to avoid division by zero ϵ_1, ϵ_2, ϵ_3 are small positive parameters. The values μ_u and ρ_u denote the mean and standard deviation of u respectively, while ρ_{uv} denotes the covariance of u and v. In practice, we take the *ssim* measure on small windows of the image, say 16×16 pixels, to obtain more local information and compute the average of this across the domain.

7.5.5 Result Sets

For clarity, we group our results by the following result sets.

Result Set 1: In this result set, we attempt to restore images Im_1-Im_6, corrupted by Gaussian blur Bl_1 without additional noise using models Mod_1-Mod_6 and New_7^1. In can be noticed in Figure 7.6 that all models perform reasonably well and are able to obtain a result in most cases.

Result Set 2: Here, we attempt to restore images Im_1-Im_6, corrupted by Gaussian blur Bl_1 and additive noise using models Mod_1-Mod_6 and New_7^1. We can see in Figure 7.7 and Tables 7.1–7.4 that all models begin to struggle to obtain a good approximation of the blur function and hence the restored images remain blurred or become over-deblurred.

Result Set 3: In this result set, we attempt to restore images Im_1-Im_6, corrupted by Gaussian blur Bl_1 and additive noise using models New_7^2 and New_7^3. We can see in

Figures 7.8–7.11 and Tables 7.1–7.4 that although New_7^2 can produce good results, it is heavily dependent on the regularisation parameter while New_7^3 is more robust and able to provide good results.

Result Set 4: In this result set, we attempt to restore images Im_1-Im_6, corrupted by out of focus blur Bl_2, box blur Bl_3 or linear motion blur Bl_4 using model New_7^3. We can see in Figures 7.12–7.14 that this model, designed with Gaussian blur in mind is not generally sufficient for dealing with other blur types.

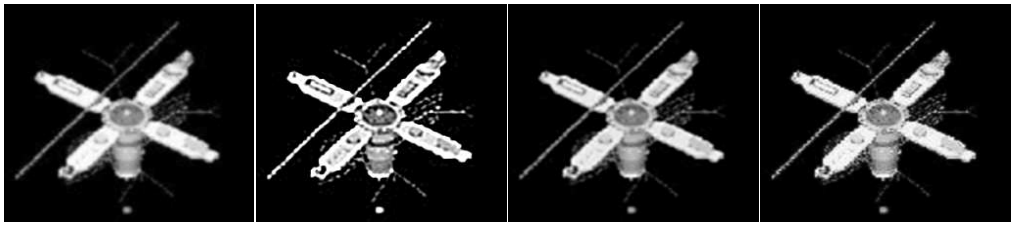
Result Set 5: In this result set, we attempt to restore images Im_1-Im_6, corrupted by out of focus blur Bl_2, box blur Bl_3 or linear motion blur Bl_4 using models New_7^4-New_7^6. We can see in Figures 7.12–7.14 that, assuming accurate identification of the blur type, we are able to recover the image using these models.

Model	Im_1		Im_2		Im_3	
	psnr	ssim	psnr	ssim	psnr	ssim
Received	25.2778	0.8191	29.1906	0.7802	26.4018	0.4565
Mod_1	25.7046	0.8238	29.0670	0.7949	27.4546	0.4648
Mod_2	16.6167	0.2052	18.2278	0.1953	26.2770	0.4663
Mod_3	25.7208	0.8239	29.2851	0.7960	27.4546	0.4648
Mod_4	25.7207	0.8239	29.7580	0.7876	19.9123	0.4516
Mod_5	25.7175	0.8193	26.5699	0.7263	24.6321	0.4577
Mod_6	25.7456	0.8239	29.8198	0.7894	25.0294	0.4592
New_7^1	26.6868	0.8261	29.8200	0.7895	27.8221	0.4670
New_7^3	46.9233	0.8285	37.9561	0.8046	37.4381	0.4969

Table 7.1: Error values for images Im_1-Im_3 corrupted by Bl_1 and a small amount of additive noise, restored by models Mod_1-Mod_6, New_7^1, New_7^3.

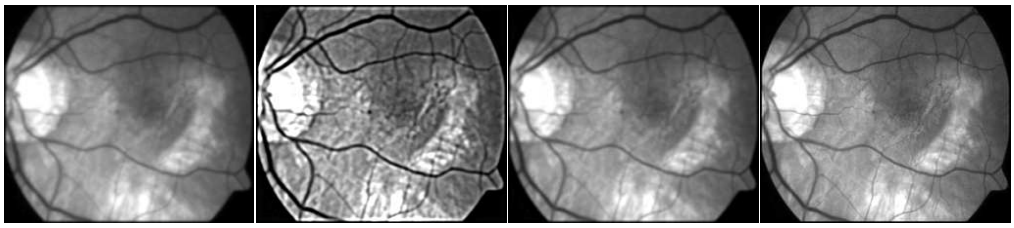
Model	Im_4		Im_5		Im_6	
	psnr	ssim	psnr	ssim	psnr	ssim
Received	22.4693	0.4014	31.4515	0.4923	22.9892	0.4056
Mod_1	20.7703	0.4168	31.0182	0.5073	23.7646	0.4211
Mod_2	17.6905	0.3953	27.0111	0.4751	23.9374	0.4274
Mod_3	22.6316	0.4272	31.1550	0.5076	23.7646	0.4211
Mod_4	23.5317	0.4203	32.2465	0.4976	24.7758	0.4626
Mod_5	23.1938	0.4228	32.3113	0.4982	21.9673	0.4254
Mod_6	23.5317	0.4203	32.2465	0.4976	23.9499	0.4244
New_7^1	23.9726	0.4270	32.2776	0.4979	25.1718	0.4635
New_7^3	34.9452	0.4966	44.7947	0.5167	35.6596	0.4991

Table 7.2: Error values for images Im_4-Im_6 corrupted by Bl_1 and a small amount of additive noise, restored by models Mod_1-Mod_6, New_7^1, New_7^3.

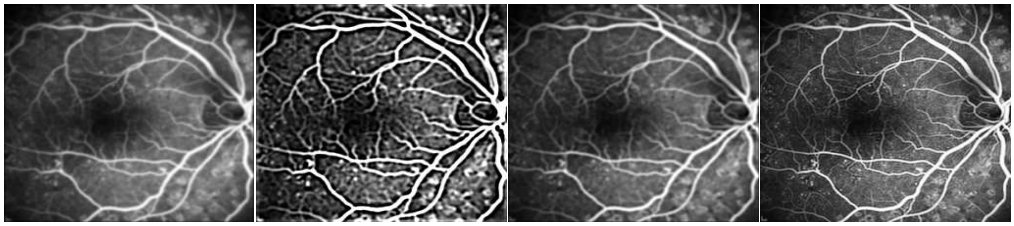


Received Image z | Mod_1, psnr=16.33 | Mod_4, psnr=31.54 | New_7^1, psnr=35.76

Received Image z | Mod_1, psnr=11.09 | Mod_4, psnr=23.97 | New_7^1, psnr=31.18



Received Image z | Mod_1, psnr=19.27 | Mod_4, psnr=33.48 | New_7^1, psnr=41.85



Received Image z | Mod_1, psnr=11.74 | Mod_4, psnr=24.52 | New_7^1, psnr=32.30

Figure 7.6: Result Set 1: Illustration of the performance of Mod_1, Mod_4 and New_7^1 with blurred images which have no additional noise. From left to right, we have 1) the received image z, 2) the restored image using Mod_1, 3) the restored image using Mod_4, and 4) the restored image using New_7^1. All models show good results while the results of New_7^1 appear to be sharper.

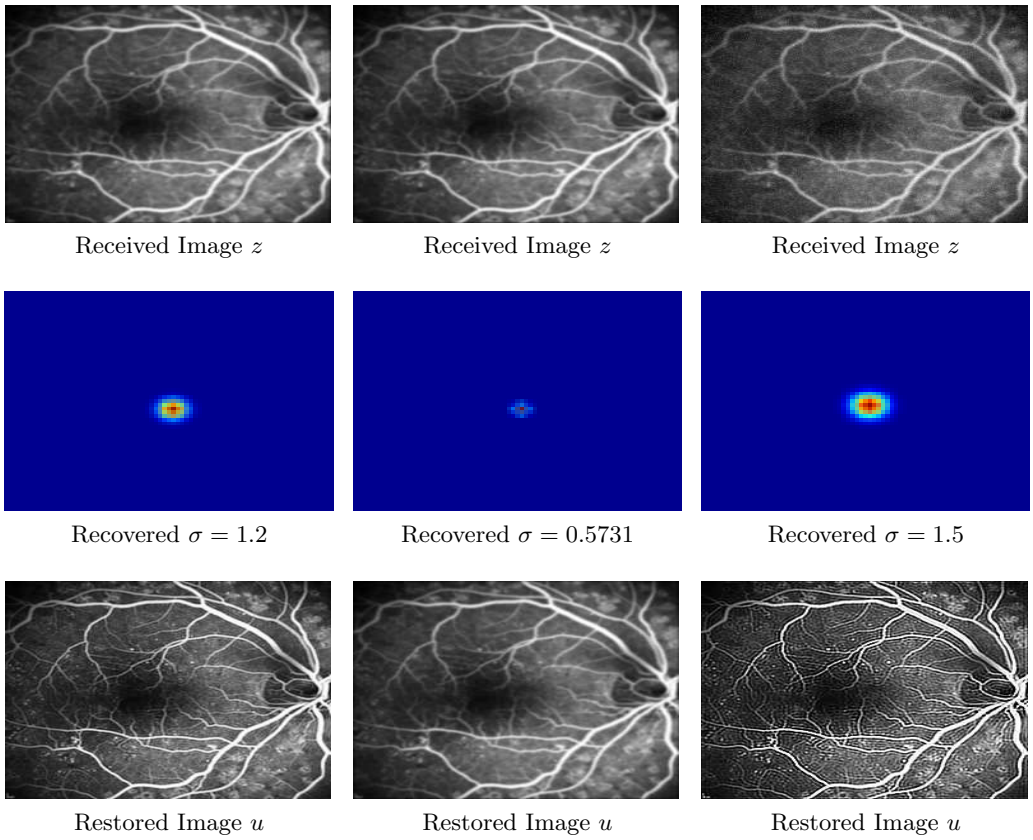
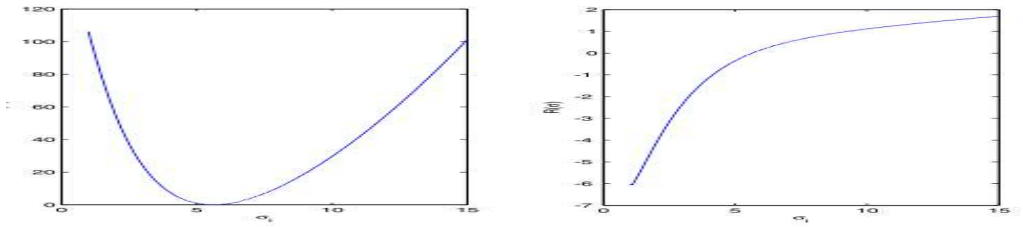


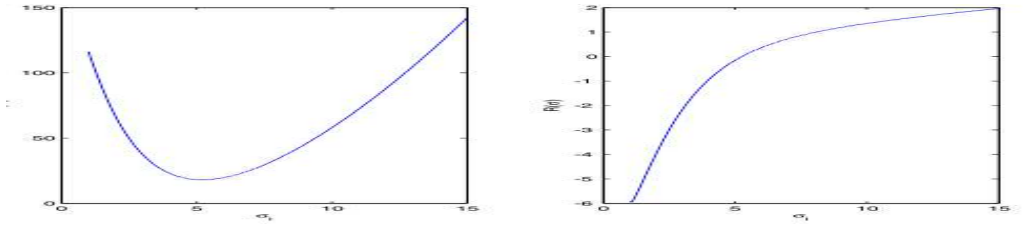
Figure 7.7: Result Set 2: Illustration of New_7^7 for Im_6. From left to right, we have 1) an example with no noise in the received image, 2) an example with noise to snr 30 in the received image, 3) an example with noise to snr 12 in the received image. From top to bottom: 1) Received image 2) Recovered blur function, 3) Recovered image. In the case of no noise, the blur function can be recovered, but in the cases of very small noise and larger noise the recovered parameters are too small leading to almost no reconstruction of the image or too large leading to over-deblurring of the image.

Model	Im_1		Im_2		Im_3	
	psnr	ssim	psnr	ssim	psnr	ssim
Received	24.9989	0.7044	28.6848	0.6963	26.1918	0.4541
Mod_1	12.2174	0.7164	19.6320	0.7247	20.1540	0.4621
Mod_2	10.1332	0.1318	10.2871	0.1192	23.3434	0.4717
Mod_3	26.7023	0.8250	20.1502	0.2209	20.1540	0.4621
Mod_4	26.0451	0.8177	29.7078	0.7954	26.6042	0.4602
Mod_5	20.0489	0.2784	13.3535	0.6131	26.0215	0.4637
Mod_6	27.9801	0.7924	30.8485	0.7935	26.6042	0.4602
New_7^1	33.5745	0.8225	31.1667	0.7950	26.8080	0.4615
New_7^3	36.0993	0.8281	37.7997	0.8045	37.4111	0.4969

Table 7.3: Error values for images Im_1-Im_3 corrupted by Bl_1 and additive noise, restored by models Mod_1-Mod_6, New_7^1, New_7^3.



σ against F_2, $\alpha = 0$ | σ against $\partial F_2/\partial\sigma$, $\alpha = 0$



σ against F_2, $\alpha = 10^{-3}$ | σ against $\partial F_2/\partial\sigma$, $\alpha = 10^{-3}$

Figure 7.8: Result Set 3: Illustration of the performance of New_7^2. On the top row, the model unsuccessful identifies $\sigma = 5.6$ for the incorrect parameter α, on the bottom row, the model successfully identifies $\sigma = 5.2$ as the minimiser of F_2 for $\alpha = 10^{-3}$. For this example, the correct reconstruction appears to rely too heavily on the choice of parameter α.

Model	Im_4		Im_5		Im_6	
	psnr	ssim	psnr	ssim	psnr	ssim
Received	22.3973	0.3975	30.4304	0.4802	22.8660	0.4026
Mod_1	19.3129	0.4365	18.9985	0.4885	18.5992	0.4224
Mod_2	22.3549	0.4332	18.9974	0.4885	23.2779	0.4131
Mod_3	19.3256	0.4365	30.3017	0.4840	18.5993	0.4224
Mod_4	22.8224	0.4103	31.0222	0.4951	23.1501	0.4111
Mod_5	22.6215	0.4155	31.0222	0.4951	23.8259	0.4218
Mod_6	22.8224	0.4103	31.1915	0.4955	23.1501	0.4111
New_7^1	23.0326	0.4134	31.9940	0.4968	23.8259	0.4511
New_7^3	34.9517	0.4966	43.4158	0.5163	35.5984	0.4990

Table 7.4: Error values for images Im_4-Im_6 corrupted by Bl_1 and additive noise, restored by models Mod_1-Mod_6, New_7^1, New_7^3.

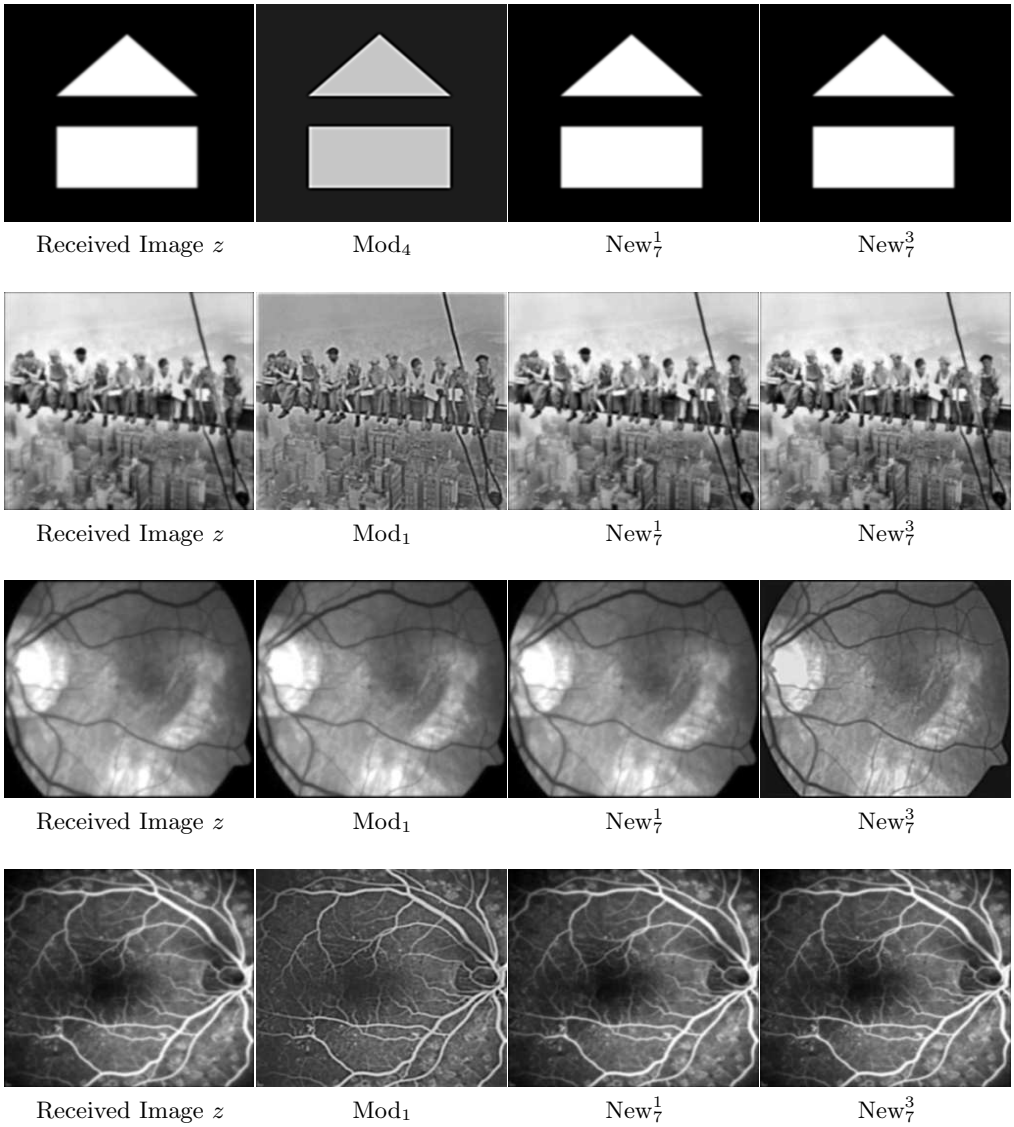


Figure 7.9: Result Set 3: Illustration of the performance of Mod_4, New_7^1, New_7^3 on blurred images with noise. From left to right, we have 1) the received image z, 2) the restored image using Mod_4, 3) the restored image using New_7^1 and 4) the restored image using New_7^3.

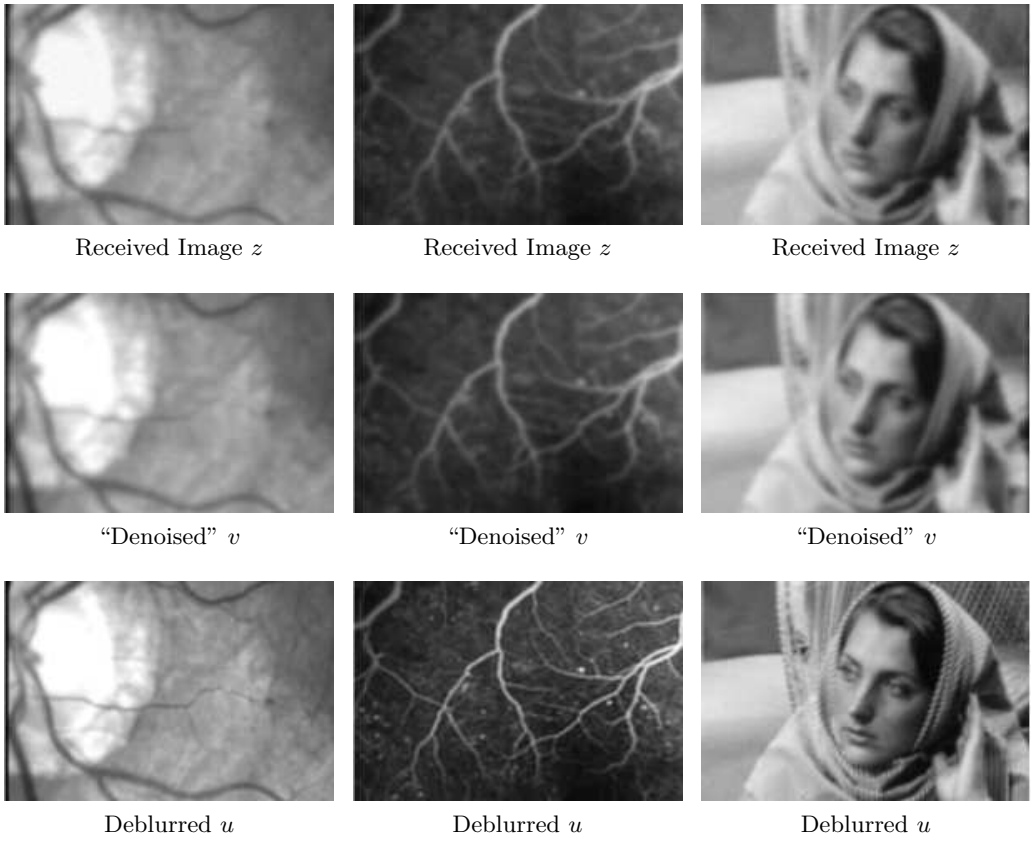


Figure 7.10: Result Set 3: Illustration of the performance of New_7^3 on noisy, blurred images. From left to right, we have 1) Im_5, 2) Im_6, 3) Im_4. From top to bottom, we have 1) the received image z, 2) the "denoised" image v and 3) the final deblurred image u. The separation of the noise from the image is successful and allows for the correct blur function to be identified and for the image to be deblurred.

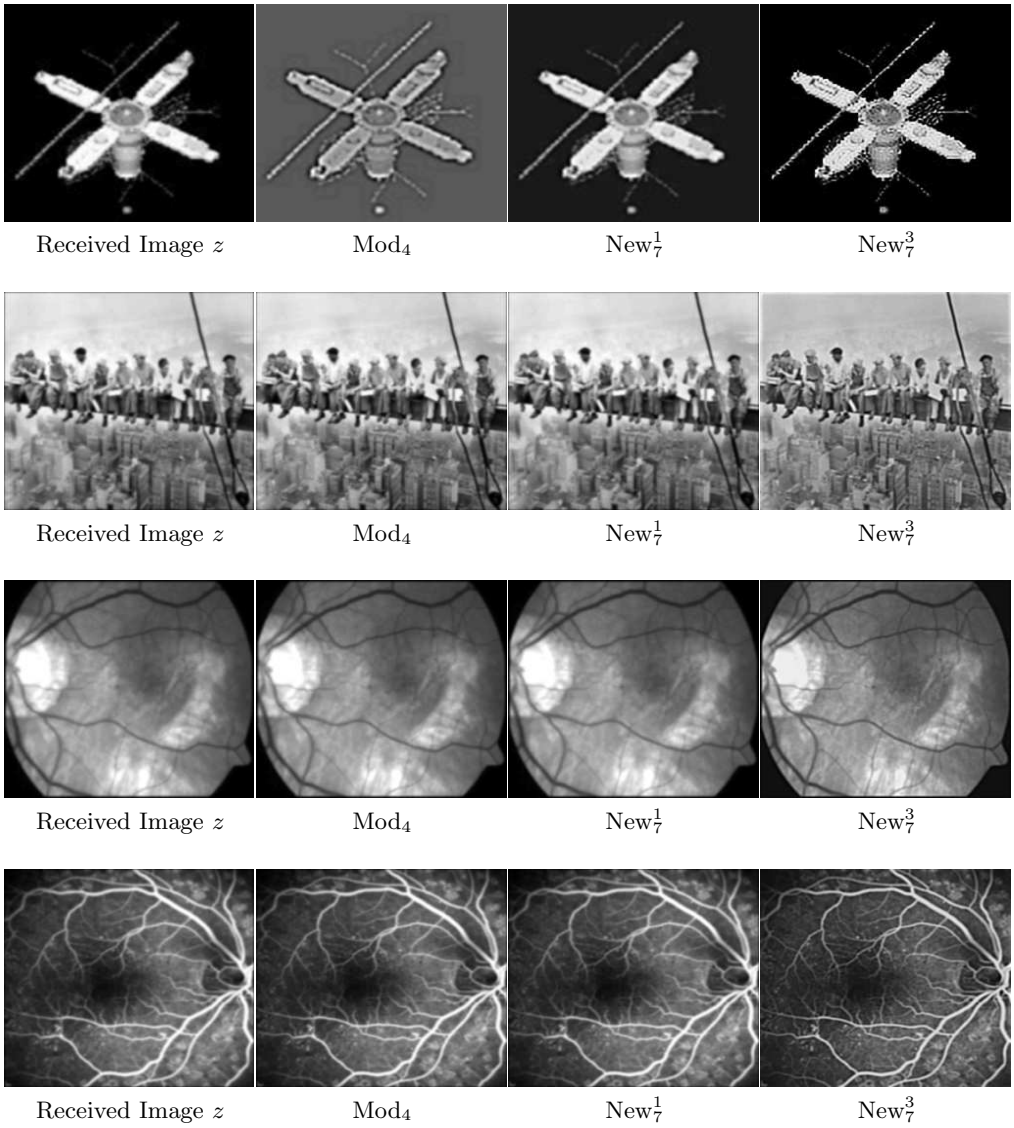


Figure 7.11: Result Set 3: Illustration of the performance of Mod_4, New_7^1 and New_7^3 on blurred images with greater noise. From left to right, we have 1) the received image z, 2) the restored image using Mod_4, 3) the restored image using New_7^1 and 4) the restored image using New_7^3.

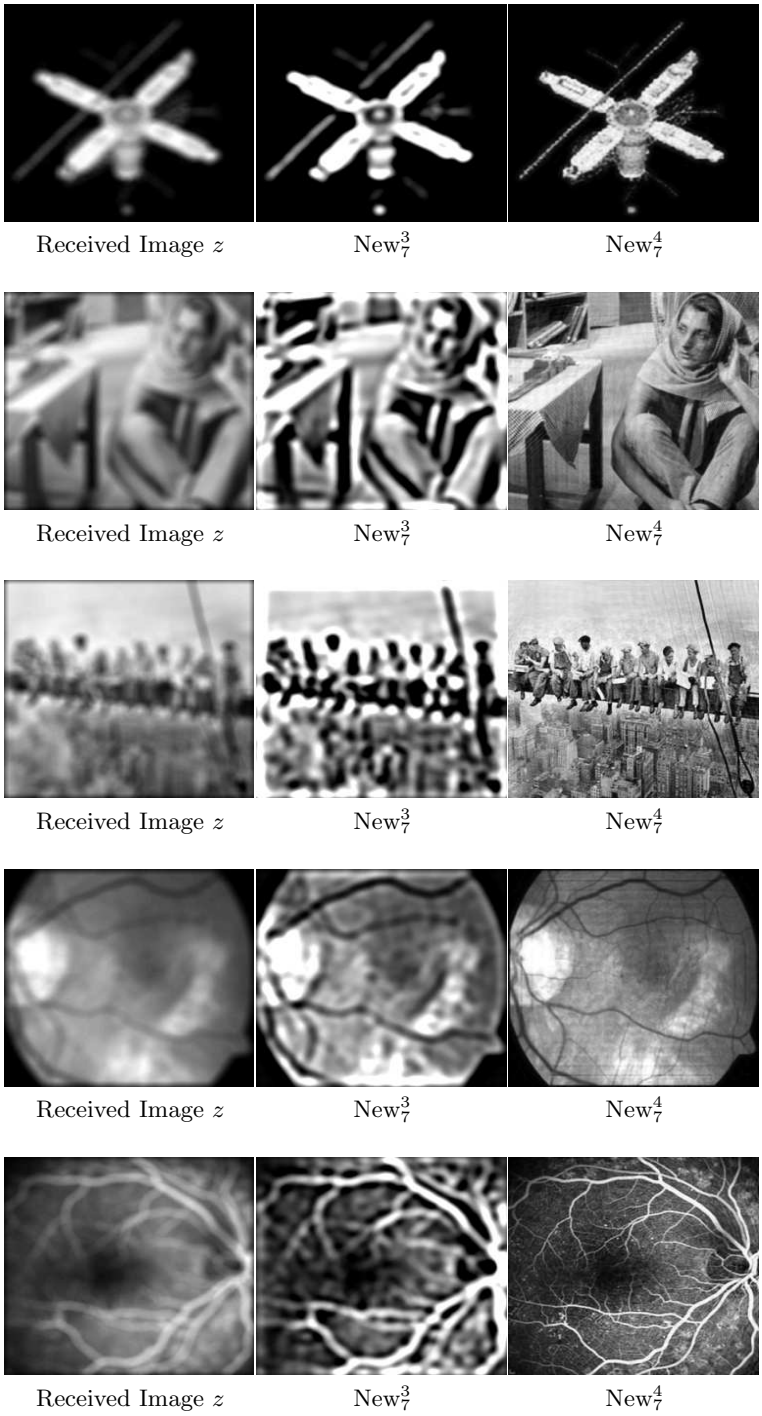


Figure 7.12: Result Sets 4 and 5: Illustration of the performance of New$_7^3$ and New$_7^4$ on images corrupted by out of focus blur. From left to right, we have 1) the received image z, 2) the restored image using New$_7^3$, 3) the restored image using New$_7^4$.

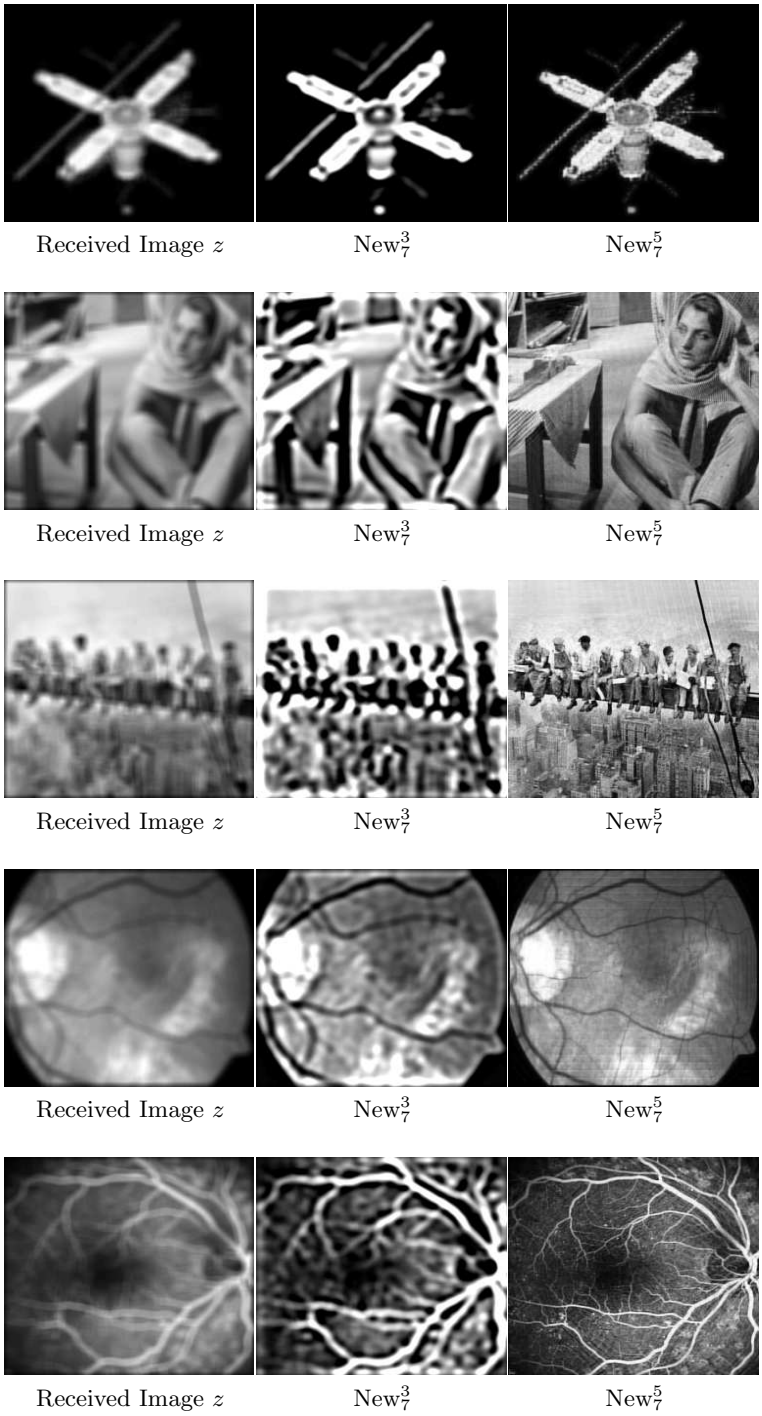


Figure 7.13: Result Sets 4 and 5: Illustration of the performance of New$_7^3$ and New$_7^5$ on images corrupted by box blur. From left to right, we have 1) the received image z, 2) the restored image using New$_7^3$, 3) the restored image using New$_7^5$.

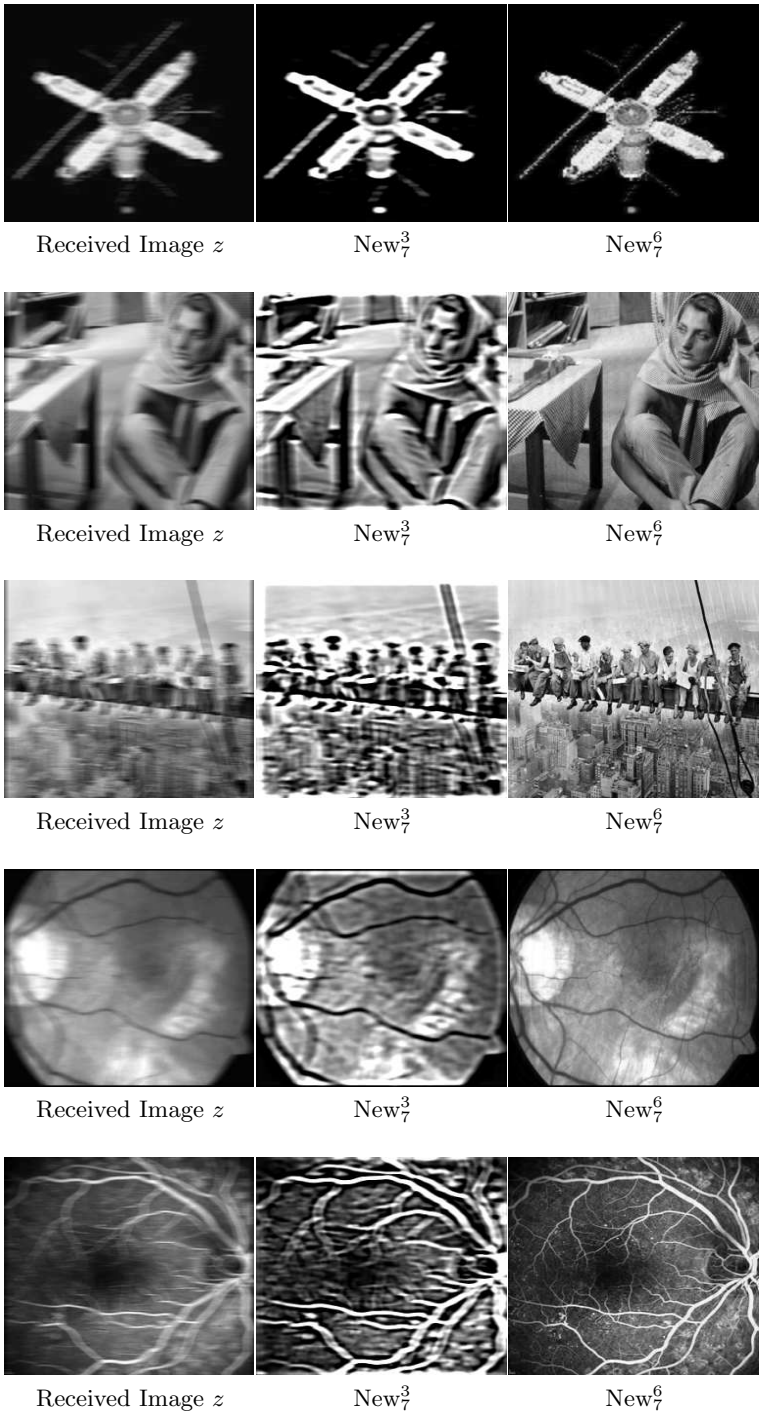


Figure 7.14: Result Sets 4 and 5: Illustration of the performance of New$_7^3$ and New$_7^6$ on images corrupted by box blur. From left to right, we have 1) the received image z, 2) the restored image using New$_7^3$, 3) the restored image using New$_7^6$.

7.6 Conclusion

We have presented a new model for parametric kernel identification and non-negative image reconstruction which offers enhanced results over existing models as well as an additional model which copes well with noise in blurred images which existing models struggle with. We have also presented formulations which allow the work to be extended beyond the case of Gaussian blur to accommodate other blur types modelled parametrically in a variational framework, which allows for significantly improved results. This model can be further developed to allow for variants of the blurring kernel, such as reconstructing the blur function as the convolution of a parametric function and a blur function recovered in the traditional non-parametric way [216, 54]. This would allow the kernel to be closely approximated and subsequently adjusted. This model may also be adapted for use with multi-channel colour images in a similar manner to [112], using inter-channel information to better inform the kernel parameter estimate, or improved using alternative regularisation.

Chapter 8

Simultaneous Reconstruction and Segmentation of Blurred Images

In this chapter, we present the fourth piece of our main work which makes full use of the new results from Chapter 6 to tackle the well-known and challenging problem of the segmentation of blurred images, which is of great importance. There have been several recent works to tackle this problem and to link the areas of image segmentation and image deconvolution. Due to lack of robust blind deconvolution methods, many existing techniques assume a known blur or a known bur type. Here we propose two variational models for simultaneous reconstruction and segmentation of blurred images with spatially invariant blur, without assuming a known blur or a known bur type.We present two solution methods for segmentation of blurred images based on implicitly constrained image reconstruction and convex segmentation. The first method is aimed at obtaining a good quality segmentation while the other is aimed at improving the speed while retaining the quality. Our results demonstrate that while existing models are capable of segmenting images corrupted by small amounts of blur they begin to struggle when faced with heavy blur degradation or noise, due to limitation of edge detectors. We demonstrate that the new algorithms are effective for segmenting blurred images in the presence of noise and offer improved results for images corrupted by strong blur.

8.1 Introduction

Image segmentation is an important technique in image processing which aims to capture either all of the objects of an image [53, 51, 49, 24, 127] or only some of them (selective segmentation [8, 162, 89, 148]). Variational models that partition images based on intensity often employ edge detection techniques to aid the segmentation and some can handle fuzzy boundaries [12, 163]. Many of these models can deal with the presence of noise, but blur proves to be more problematic and most variational models struggle to capture all of the required objects, especially in cases where there is a reliance on the edge detector.

Work in the segmentation of blurred images is at an early stage but there exist models such as those presented in [10, 45, 166, 111] which aim to segment blurred images based on Mumford-Shah [143, 53, 4, 113] or Chan-Vese [63, 62, 53] segmentation and total variation image restoration [169, 193]. In [10], the authors attempt to segment blurred images by forming a joint functional incorporating segmentation and image reconstruction, assuming the blur type is known (in the so-called semi-blind method). A framework of alternate minimisation is adopted such that the image is simultaneously restored and segmented. Similar techniques can also be found in other works such as [111]. As an alternative approach, some models such as those given in [45, 166] treat the problems of image reconstruction and segmentation separately in a two-stage model by first restoring the image and then segmenting the restored data.

The main contribution of this work is the proposal of two new models which incorporate blind deconvolution (with implicitly constrained image reconstruction) and convex segmentation. Here the former task is particularly important in astronomical imaging and medical imaging and offers advantages over hard constraints such as scaling or truncation, while the latter provides a global minimiser to the segmentation problem with no reliance on the initial guess of objects.In a similar manner to [10], we form a joint functional and proceed to segment the image using alternate minimisation, although it is feasible to use a two-stage approach out of the joint model (as tested and compared later). We also present an accelerated solution method for speed-up while sacrificing only a small amount of quality.

The rest of this chapter is organised as follows. In §8.2, we build on the review of segmentation in §3.5 and consider efforts to tackle the problem of segmenting images which have been corrupted by blur. In §8.3, we introduce 2-stage models for the cases of deblurring in the presence of Gaussian noise and Poisson noise. In §8.4, we introduce our new joint model for segmentation of blurred images. In §8.5, we introduce our accelerated model for segmentation of blurred images. In §8.6, we present experimental results. In §8.7, we present the conclusions of this work.

8.2 Existing Methods

In recent years, several approaches have been developed to tackle the problem of accurately segmenting images which have been corrupted by blur. Such approaches most commonly involve image reconstruction to restore the "true image" and segmentation. Models can be classed as *two-stage* in which reconstruction of the corrupted image is carried out, followed by segmentation of the restored image [45, 166]. In contrast, there also exist *joint* models which attempt to deal with the tasks of reconstruction and segmentation simultaneously by minimising a joint functional [10, 111].

Bar et al. showed in their 2004 paper [10] that the two problems of segmentation and image restoration could be coupled together and hence solved at the same time. Both of the case of non-blind deconvolution where the kernel is known and the case of

semi-blind deconvolution assuming that the blur function is of Gaussian type, leaving the width σ of the Gaussian function describing the blur to be found, are considered. The problem was solved by minimising the joint functional

$$\begin{aligned} f_{BSK}(u, h_\sigma, v) =& \frac{1}{2}\int_\Omega (h_\sigma * u - z)^2 \, \mathrm{d}A + \beta \int_\Omega v^2 |\nabla u|^2 \, \mathrm{d}A \\ &+ \alpha \int_\Omega \varepsilon |\nabla v|^2 + \frac{(v-1)^2}{4\varepsilon} \, \mathrm{d}A + \gamma \int_\Omega |\nabla h_\sigma|^2 \, \mathrm{d}A, \end{aligned}$$

dependent on the image u, the edge integration map v and the kernel function parameter σ. Minimising with respect to the arguments, the authors simultaneously recover and segment the image. A special case exists in the case of known blur where minimisation with respect to the kernel width is not necessary, nor is the final term of the functional.

Jung et al. [111] present models for joint multi-phase segmentation, deblurring and denoising of images by considering the region based active contours without edges model and give the joint formulation as the minimisation of the energy functional

$$f_{JCSV}(c_1, c_2, \phi) = \int_\Omega |z - k * (c_1 H(\phi) + c_2(1 - H(\phi)))|^2 \, \mathrm{d}x + \mu \int_\Omega |\nabla H(\phi)| \, \mathrm{d}x$$

for two phase segmentation. A similar model is also presented to allow for the segmentation into m distinct levels. This is then solved by alternate minimisation of the arguments, solving the H^1 gradient descent $\phi_t(x,t) = -\nabla_{H^1} f_{JCSV}(c_1, c_2, \phi)$.

Reddy et al. [166] approach this problem in the blind case, where the blur function is not known using the Chan-Vese snake model

$$\begin{aligned} f_{RCR} =& \mu \mathrm{Length}(c) + \nu \mathrm{Area}(\mathrm{inside}(c) \\ &+ \lambda_1 \int_{\mathrm{inside}(c)} |u(x,y) - c_1|^2 \, \mathrm{d}\vec{x} + \lambda_2 \int_{\mathrm{outside}(c)} |u(x,y) - c_2|^2 \, \mathrm{d}\vec{x} \end{aligned}$$

where $u(x,y) = |h(x,y) * z(x,y)|^2$ is the square of the convolution of the received data z and a Gaussian kernel, c_1 and c_2 are constants which approximate the image intensities inside and outside of the contour C, and $\mu, \nu, \lambda_1, \lambda_2$ are non-negative parameters. The authors adopt a two stage approach, first applying a deblurring algorithm, followed by segmentation of the reconstructed image.

Chan et al. [45] recently presented a two stage *convex* method for segmenting blurred images which have been corrupted by either Poisson or multiplicative Gamma noise. Their technique is to extract a smooth image u from the received image z and then to threshold u to reveal segmentation features. The functional to be minimised, given the blurring operator $\mathcal{A}$, is

$$f_{CYZ}(u) = \int_\Omega |\nabla u| \, \mathrm{d}x + \frac{\mu}{2} \int_\Omega |\nabla u|^2 \, \mathrm{d}x + \lambda \int_\Omega \mathcal{A}u - f \log \mathcal{A}u \, \mathrm{d}x$$

which has a unique solution and can be solved by split-Bregman [86] or Chambolle-Pock [41].

In this chapter we shall consider both joint and two-stage approaches using convex segmentation and implicitly constrained deblurring in an attempt to improve the quality of the results.

8.2.1 Constrained Image Reconstruction

Total variation (TV) deblurring is widely used [11, 103, 104, 149, 152, 169] due to its success in recovering sharp images but, while unconstrained TV deblurring provides very good results, the restored image may contain values significantly outside of the boundary of the intensity values meaning that it must be projected back onto the correct range. This is typically done in a naive way which can lead to a significant drop in the quality of the recovered image [44]. To address this, we implement a model for non-negative implicitly constrained deblurring which is known to provide a better result than unconstrained deblurring.

To this end, we introduce a transform on the image given by $\tau_{\boldsymbol{a}}(\psi)$ where $\tau_{\boldsymbol{a}} : \mathbb{R}^{m\times n} \to \mathcal{C}, \mathcal{C} \subseteq \mathbb{R}^{m\times n}$ is a function which has a range contained in $\mathcal{C}$ and $\psi : \Omega \to \mathbb{R}^{m\times n}$ is a function such that $\tau_{\boldsymbol{a}}(\psi)$ is equal to the image u. We have found that a bounded transform may be given by

$$\tau_{\boldsymbol{a}}(\psi) = \frac{a_1 + 2a_4}{1 + a_2 e^{-\frac{2\psi}{a_3}}} - a_4$$

where the parameters $a_1, \ldots, a_4$ may be preselected and are problem-dependent. Other choices of $\tau_{\boldsymbol{a}}(\psi)$ may be permitted including the somewhat easy but unbounded transform $\tau_{\boldsymbol{a}}(\psi) = \psi^2$. The four parameters may be fixed and they allow some flexibility, each attempting to constrain some aspect of the transform. For brevity, we consider two of the parameters: a_1 should be equal to the maximal expected intensity of the image and a_4 gives the tolerance of recovered intensities which may fall outside of the natural range. Both of these may be given by inspection of the received data. To give an example, for an image whose intensity values are contained in the range $[0, 1]$, an appropriate parameter choice can be $\boldsymbol{a} = \left\{1, 1, 0.44, 10^{-2}\right\}$. The inverse transform $\xi : \mathcal{C} \to \mathbb{R}^{m\times n}$ is given such that $\psi = \xi_{\boldsymbol{a}}(u)$. To avoid introducing non-linearity into the fitting term $(k * u - z)$ by the transform, we use the augmented Lagrangian method [44, 198, 188, 197, 104] to retain the variable u in the fitting term and, through $|u - \tau_{\boldsymbol{a}}(\psi)|$, to drive u and $\tau_{\boldsymbol{a}}(\psi)$ close to each other. Applying similar considerations to the point spread function, we have the non-negative implicitly constrained model given by

$$\begin{aligned}\max_{\varphi_1,\varphi_2} \min_{u,\psi,k,\omega} \Big\{ f_{CTVK}(u, \psi, k, \omega; \varphi_1, \varphi_2) = \frac{1}{2} \|k * u - z\|^2_{L^2(\Omega)} + \alpha_1 \int_\Omega |\nabla \tau_{\boldsymbol{a}}(\psi)| \\ + \frac{\gamma_1}{2} \|u - \tau_{\boldsymbol{a}}(\psi)\|^2_{L^2(\Omega)} + < \varphi_1, u - \tau_{\boldsymbol{a}}(\psi) > + \alpha_2 \int_\Omega |\nabla \tau_{\boldsymbol{b}}(\omega)| \\ + \frac{\gamma_2}{2} \|k - \tau_{\boldsymbol{b}}(\omega)\|^2_{L^2(\Omega)} + < \varphi_2, k - \tau_{\boldsymbol{b}}(\omega) > \Big\}. \end{aligned} \tag{8.1}$$

Note that if we can give the point spread function precisely then we can relax this equation to achieve the non-blind case by fixing $\alpha_2 = \gamma_2 = 0$ and $\varphi_2 = \mathbf{0}$. We remark that, instead of TV regularisation, we might improve this estimate with alternative regularisation [188, 71, 153] such as total generalised variation [23, 116], nonlocal TV [213, 217] or high order methods [26, 56, 133] but since optimising this choice is not our aim, we use mainly TV.

8.3 Two-Stage Models for Restoring and Segmenting Blurred Images

Returning to the problem of segmenting blurred images, we first consider the idea of carrying out this task within a 2-stage model. That is, we first aim to reconstruct the sharp image from the corrupted received data and proceed to segment the result. This is a popular formulation and has been used in work such as [166, 45].

In the case of assuming that Gaussian noise exists in the received data, we first solve the problem given in Equation (8.1) in order to obtain the deblurred image denoted by u. Assuming that we know the blur function and carrying out the minimisation with respect to the arguments in the non-blind case, we obtain the Euler Lagrange equations

$$k^\dagger * (k * u - z) + \gamma_1 (u - \tau_{\boldsymbol{a}}(\psi)) + \varphi_1 u = 0, \tag{8.2}$$

$$\alpha_1 \frac{\partial \tau_{\boldsymbol{a}}}{\partial \psi} \nabla \cdot \left(\frac{\nabla \tau_{\boldsymbol{a}}(\psi)}{|\nabla \tau_{\boldsymbol{a}}(\psi)|} \right) + \gamma_1 (\tau_{\boldsymbol{a}}(\psi) - u) \frac{\partial \tau_{\boldsymbol{a}}}{\partial \psi} + \varphi_1 (\tau_{\boldsymbol{a}}(\psi) - u) \frac{\partial \tau_{\boldsymbol{a}}}{\partial \psi}. \tag{8.3}$$

By alternately minimising (8.1) we may obtain an approximation of the true image given by u. We then proceed to segment this restored image by solving Equation (3.57), replacing the function z with the restored image u:

$$\min_{\phi, c_1, c_2} \left\{ \mu \int_\Omega |\nabla H(\phi)| + \lambda_1 \int_\Omega |u - c_1|^2 H(\phi) \, \mathrm{d}\Omega + \lambda_2 \int_\Omega |u - c_2|^2 (1 - H(\phi)) \, \mathrm{d}\Omega \right\}. \tag{8.4}$$

In this way, we can obtain a segmentation of the blurred image.

This formulation can however lead to the solution of ϕ being a local minimum, due to the non-convexity of the original Mumford-Shah equation (3.56), which is a binary labelling problem. In order to guarantee finding a global minimum independent of initialisation, we employ the method of convex relaxation, which we detail below. Originally, this idea was applied to two-phase, piecewise constant segmentation (3.56) in [49]. Further work [24, 161, 85] has been done in recent years, and the framework extended to multiphase segmentation [27, 122]. Recently, convex relaxation has been applied to segmentation with shape priors [115] and selective segmentation [148]. In (8.11) the Heaviside function can be represented by an indicator function ν, that is 1 inside the contour Γ and 0 outside. By relaxing this binary constraint, allowing ν to

take intermediate values, we get the constrained minimisation problem

$$\min_{0\leq\nu\leq 1}\left\{\mu\int_\Omega|\nabla\nu|+\lambda_1\int_\Omega|u-c_1|^2\nu\,\mathrm{d}\Omega+\lambda_2\int_\Omega|u-c_2|^2(1-\nu)\,\mathrm{d}\Omega\right\}. \tag{8.5}$$

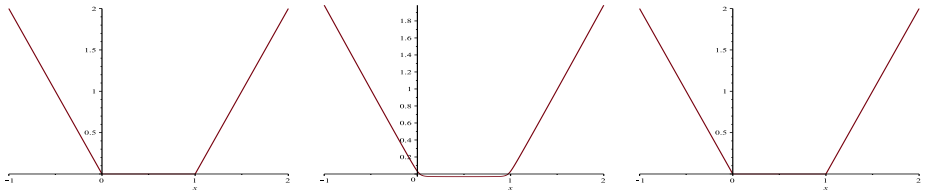
To enforce the constraint $\nu\in[0,1]$, we use the penalty function $\varsigma(\nu)=\max\{0,2|\nu-\frac{1}{2}|-1\}$ used in [49], weighted by a parameter, σ. The term $\sigma\varsigma(\nu)$ is an exact penalty term [100] and modifies (8.5) to an unconstrained problem, which has the same set of minimisers provided that

$$\sigma>\|\ \tfrac{\lambda_1}{2}|u-c_1|^2-\tfrac{\lambda_2}{2}|u-c_2|^2\ \|_{L^\infty(\Omega)},$$

with a proof provided in [49]. We introduce a regularised version of $\varsigma(\nu)$ given by

$$\varsigma_\varepsilon(\nu)=b_\varepsilon(\nu)H_\varepsilon\left(b_\varepsilon(\nu)\right),\qquad b_\varepsilon(x)=\sqrt{(2x-1)^2+\varepsilon}-1, \tag{8.6}$$

where H_ε is the Heaviside approximation defined above. When computing the global minimiser, ν^*, thresholding the function at any value $p\in(0,1)$ gives the contour of the object, Γ_p.



$\varsigma(x)$ $\qquad$ $\varsigma_\varepsilon(x),\ \varepsilon=10^{-2}$ $\qquad$ $\varsigma_\varepsilon(x),\ \varepsilon=10^{-3}$

Figure 8.1: Illustration of the continuous approximation ς_ε to the piecewise linear function ς. For lower ε, the approximation is very close to ς.

Incorporating this idea into our 2-stage approach, we obtain the segmentation of the restored image by solving the problem

$$\min_{0\leq\nu\leq 1}\left\{\mu\int_\Omega|\nabla\nu|+\lambda_1\int_\Omega|u-c_1|^2\nu\,\mathrm{d}\Omega+\lambda_2\int_\Omega|u-c_2|^2(1-\nu)\,\mathrm{d}\Omega\right\} \tag{8.7}$$

and enforce the constraint $\nu\in[0,1]$ using the penalty function $\varsigma_\varepsilon(\nu)$ given above in (8.6). We then obtain the contour of the object by thresholding the global minimiser ν^* at a value $p\in(0,1)$. Incorporating this idea into (8.11), we present the new joint model as

$$\begin{aligned}\min_{\nu,c_1,c_2}\Big\{\ &\mu\int_\Omega|\nabla\nu|\,\mathrm{d}\Omega+\lambda_1\int_\Omega|u-c_1|^2\nu\,\mathrm{d}\Omega\\ &+\lambda_2\int_\Omega|u-c_2|^2(1-\nu)\,\mathrm{d}\Omega+\sigma\int_\Omega b_\varepsilon H_\varepsilon\left(b_\varepsilon\right)\mathrm{d}\Omega\Big\},\end{aligned} \tag{8.8}$$

where $b_\varepsilon=b_\varepsilon(\nu)$. We derive the Euler Lagrange equation by minimising with respect

to ν given by

$$\mu\nabla\cdot\left(\frac{\nabla\nu}{|\nabla\nu|}\right)+\lambda_1(u-c_1)^2-\lambda_2(u-c_2)^2+\frac{2\sigma(2\nu-1)}{b_\varepsilon+1}\left(H_\varepsilon\left(b_\varepsilon\right)\delta_\varepsilon\left(b_\varepsilon\right)b_\varepsilon\right)=0, \quad (8.9)$$

where $\delta_\epsilon(\nu)$ is a smooth approximation of the delta function. Keeping the other arguments fixed and minimising with respect to c_1 and c_2, we have respectively the equations given below:

$$c_1(u,\nu)=\frac{\int_\Omega u\,\nu\,\mathrm{d}\Omega}{\int_\Omega \nu\,\mathrm{d}\Omega}, \qquad c_2(u,\nu)=\frac{\int_\Omega u(1-\nu)\,\mathrm{d}\Omega}{\int_\Omega (1-\nu)\,\mathrm{d}\Omega}. \quad (8.10)$$

In order to solve this model, we make an initial estimate of the image, which is typically the received data since it's the closest approximation we have. Using this information, we obtain the approximation u of the true image and using this we proceed with alternate minimisation of (8.8) until we reach an acceptable tolerance. Our algorithm is presented in Algorithm 21.

Algorithm 21 Segmentation of Blurred Images (wsc_1)

function $\nu \leftarrow \text{WSC}_1(\nu^{(0)}, z, maxit)$
  $u^{(0)} \leftarrow z$, $\psi^{(0)} \leftarrow \xi_{\boldsymbol{a}}\left(u^{(0)}\right)$
  $\varphi_1^{(0)} \leftarrow \mathbf{1}$
  for $\ell \leftarrow 1 : maxit$ **do**
    Update $u^{(\ell)}$ using (8.2)
    Update $\psi^{(\ell)}$ using (8.3) and time marching
    Update $\varphi_1^{(\ell)} \leftarrow \varphi_1^{(\ell-1)} + \gamma_1\left(u^{(\ell)} - \tau_{\boldsymbol{a}}\left(\psi^{(\ell)}\right)\right)$
  end for
  $u \leftarrow \tau_{\boldsymbol{a}}\left(\psi^{\ell}\right)$
  for $\ell \leftarrow 1 : maxit$ **do**
    Calculate $c_1^{(\ell)} \leftarrow c_1\left(u, \nu^{(\ell-1)}\right)$, $c_2^{(\ell)} \leftarrow c_2\left(u, \nu^{(\ell-1)}\right)$ using (8.10)
    Update $\nu^{(\ell)}$ using (8.9) and AOS
  end for
  $\nu \leftarrow \nu^{(\ell)}$
end function

Considering now the case of Poisson noise being present in the image, we make an alteration to our deblurring algorithm to take this into account, as discussed in §3.3.7. We thus attempt to restore the true image from the corrupted image by solving the Robust Richardson Lucy problem given by Equation (3.35), employing the function $\Phi(s) = 2\sqrt{s+\beta}$ [157]. Solving this problem using the iterative scheme given by (3.36), we obtain an approximation of the true image. In this two-stage setting, once we have obtained the approximation of the image, we proceed with the segmentation. This is outlined in Algorithm 22 below. It can be noted that while this restoration method provides a restriction on the lower bound of the restored image u, it does not provide an upper limit. We may obtain this by a projection $\mathcal{P}$ of the restored data onto the ideal range at each iteration.

Algorithm 22 Segmentation of Blurred Images (wsc_2)

function $\nu \leftarrow \text{WSC}_2(\nu^{(0)}, z, maxit)$
  $u^{(0)} \leftarrow z$
  for $\ell \leftarrow 1 : maxit$ **do**
    Update $u^{(\ell)}$ using (3.36)
  end for
  $u \leftarrow \tau_{\boldsymbol{a}}\left(\psi^{\ell}\right)$
  for $\ell \leftarrow 1 : maxit$ **do**
    Calculate $c_1^{(\ell)} \leftarrow c_1\left(u, \nu^{(\ell-1)}\right)$, $c_2^{(\ell)} \leftarrow c_2\left(u, \nu^{(\ell-1)}\right)$ using (8.10)
    Update $\nu^{(\ell)}$ using (8.9) and AOS
  end for
  $\nu \leftarrow \nu^{(\ell)}$
end function

8.4 A New Joint Model for Simultaneous Segmentation and Deblurring

We now construct a joint variational model for simultaneous image reconstruction and segmentation. Then, by minimisation of a single energy functional, the image is simultaneously restored and segmented. We form the joint model from replacing the data term z in (3.57) by the transform function $\tau_{\boldsymbol{a}}(\psi)$ which is essentially u and should satisfy (8.1). We present this joint model as

$$\begin{aligned}\max_{\varphi_1,\varphi_2}\min_{\phi,c_1,c_2,u,\psi,k,\omega}\Big\{ & f(\phi,c_1,c_2,u,\psi,k,\omega;\varphi_1,\varphi_2) = \mu\int_\Omega |\nabla H(\phi)| \\ & + \lambda_1\int_\Omega |\tau_{\boldsymbol{a}}(\psi)-c_1|^2 H(\phi)\,\mathrm{d}\Omega + \lambda_2\int_\Omega |\tau_{\boldsymbol{a}}(\psi)-c_2|^2(1-H(\phi))\,\mathrm{d}\Omega \\ & + \frac{1}{2}\|k*u-z\|^2_{L^2(\Omega)} + \alpha_1\int_\Omega |\nabla\tau_{\boldsymbol{a}}(\psi)| + \frac{\gamma_1}{2}\|u-\tau_{\boldsymbol{a}}(\psi)\|^2_{L^2(\Omega)} \\ & + <\varphi_1, u-\tau_{\boldsymbol{a}}(\psi)> + \alpha_2\int_\Omega |\nabla\tau_{\boldsymbol{b}}(\omega)| + \frac{\gamma_2}{2}\|k-\tau_{\boldsymbol{b}}(\omega)\|^2_{L^2(\Omega)} \\ & + <\varphi_2, k-\tau_{\boldsymbol{b}}(\omega)> \Big\}. \end{aligned} \tag{8.11}$$

Similar to (8.4), the solution of ϕ could be a local minimum. In order to obtain a global minimum, we again employ the method of convex relaxation. We replace the Heaviside function in (8.11) with an indicator function ν equal to one inside the contour and zero outside. Relaxing this constraint, we obtain the constrained minimisation problem

$$\min_{0\le\nu\le 1}\left\{\mu\int_\Omega |\nabla\nu| + \lambda_1\int_\Omega |\tau_{\boldsymbol{a}}(\psi)-c_1|^2\nu\,\mathrm{d}\Omega + \lambda_2\int_\Omega |\tau_{\boldsymbol{a}}(\psi)-c_2|^2(1-\nu)\,\mathrm{d}\Omega\right\} \tag{8.12}$$

and enforce the constraint $\nu \in [0,1]$ using the penalty function $\varsigma_\varepsilon(\nu)$ given above in (8.6). We then obtain the contour of the object by thresholding the global minimiser ν^* at a value $p \in (0,1)$. Incorporating this idea into (8.11), we present the new joint

model as

$$\max_{\varphi_1,\varphi_2}\min_{\nu,c_1,c_2,u,\psi,k,\omega}\Big\{f_{\text{wsc}_3}(\nu,c_1,c_2,u,\psi,k,\omega;\varphi_1,\varphi_2)=\mu\int_\Omega|\nabla\nu|$$
$$+\lambda_1\int_\Omega|\tau_{\boldsymbol{a}}(\psi)-c_1|^2\nu\,\mathrm{d}\Omega+\lambda_2\int_\Omega|\tau_{\boldsymbol{a}}(\psi)-c_2|^2(1-\nu)\,\mathrm{d}\Omega$$
$$+\sigma\int_\Omega b_\varepsilon(\nu)H_\varepsilon\left(b_\varepsilon(\nu)\right)\mathrm{d}\Omega+\frac{1}{2}\left\|k*u-z\right\|^2_{L^2(\Omega)}+\alpha_1\int_\Omega|\nabla\tau_{\boldsymbol{a}}(\psi)|$$
$$+\frac{\gamma_1}{2}\left\|u-\tau_{\boldsymbol{a}}(\psi)\right\|^2_{L^2(\Omega)}+<\varphi_1,u-\tau_{\boldsymbol{a}}(\psi)>+\alpha_2\int_\Omega|\nabla\tau_{\boldsymbol{b}}(\omega)|$$
$$+\frac{\gamma_2}{2}\left\|k-\tau_{\boldsymbol{b}}(\omega)\right\|^2_{L^2(\Omega)}+<\varphi_2,k-\tau_{\boldsymbol{b}}(\omega)>\Big\}. \tag{8.13}$$

In order to solve this model, we derive the partial differential equations defined by the first order optimality conditions. We will take each of the arguments in turn. Minimising with respect to ν, we derive the Euler Lagrange equation from $\partial f_{\text{wsc}_3}/\partial\nu$:

$$\mu\nabla\cdot\left(\frac{\nabla\nu}{|\nabla\nu|}\right)+\lambda_1(\tau_a(\psi)-c_1)^2-\lambda_2(\tau_a(\psi)-c_2)^2$$
$$+\frac{2\sigma(2\nu-1)}{b_\varepsilon(\nu)+1}\left(H_\varepsilon\left(b_\varepsilon(\nu)\right)\delta_\varepsilon\left(b_\varepsilon(\nu)\right)b_\varepsilon(\nu)\right)=0, \tag{8.14}$$

where $\delta_\epsilon(\nu)$ is a smooth approximation to the delta function. Keeping the other arguments fixed and minimising with respect to c_1 and c_2, we have respectively the equations given by $\partial f_{\text{wsc}_3}/\partial c_1$ and $\partial f_{\text{wsc}_3}/\partial c_2$:

$$c_1(\psi,\nu)=\frac{\int_\Omega\tau_{\boldsymbol{a}}(\psi)\nu\,\mathrm{d}\Omega}{\int_\Omega\nu\,\mathrm{d}\Omega},\qquad c_2(\psi,\nu)=\frac{\int_\Omega\tau_{\boldsymbol{a}}(\psi)(1-\nu)\,\mathrm{d}\Omega}{\int_\Omega(1-\nu)\,\mathrm{d}\Omega}. \tag{8.15}$$

Minimising now with respect to u, we have the equation

$$k^\dagger*(k*u-z)+\gamma_1(u-\tau_{\boldsymbol{a}}(\psi))+\varphi_1(u-\tau_{\boldsymbol{a}}(\psi))=0, \tag{8.16}$$

and minimising with respect to ψ, we obtain:

$$2\lambda_1(\tau_{\boldsymbol{a}}(\psi)-c_1)\nu\frac{\partial\tau_{\boldsymbol{a}}(\psi)}{\psi}+2\lambda_2(\tau_{\boldsymbol{a}}(\psi)-c_2)(1-\nu)\frac{\partial\tau_{\boldsymbol{a}}(\psi)}{\psi}$$
$$+\alpha_1\frac{\partial\tau_{\boldsymbol{a}}(\psi)}{\psi}\nabla\cdot\left(\frac{\nabla\tau_{\boldsymbol{a}}(\psi)}{|\nabla\tau_{\boldsymbol{a}}(\psi)|}\right)-\gamma_1(u-\tau_{\boldsymbol{a}}(\psi))\frac{\partial\tau_{\boldsymbol{a}}(\psi)}{\psi}$$
$$-\varphi_1(u-\tau_{\boldsymbol{a}}(\psi))\frac{\partial\tau_{\boldsymbol{a}}(\psi)}{\psi}=0. \tag{8.17}$$

Minimising now with respect to k, we have the equation

$$u^\dagger*(u*k-z)+\gamma_2(k-\tau_{\boldsymbol{b}}(\omega))+\varphi_2(k-\tau_{\boldsymbol{b}}(\omega))=0, \tag{8.18}$$

and minimising with respect to ω, we obtain:

$$\alpha_2 \frac{\partial \tau_{\boldsymbol{b}}(\omega)}{\omega} \nabla \cdot \left(\frac{\nabla \tau_{\boldsymbol{b}}(\omega)}{|\nabla \tau_{\boldsymbol{b}}(\omega)|} \right) - \gamma_2 (k - \tau_{\boldsymbol{b}}(\omega)) \frac{\partial \tau_{\boldsymbol{b}}(\omega)}{\omega} - \varphi_2 (k - \tau_{\boldsymbol{b}}(\omega)) \frac{\partial \tau_{\boldsymbol{b}}(\omega)}{\omega} = 0. \tag{8.19}$$

In order to solve this model, we make an initial estimate of the image, which is typically the received data since it's the closest approximation we have. We also make an estimate of the point spread function based on visual observation of the received image. Using this information, we obtain the initial estimates of ψ and ω, then calculate the first estimates of c_1 and c_2. We next update the image, ψ, the point spread function, ω, the function ν, φ_1 and φ_2. We proceed to iterate until we reach an acceptable tolerance. Our algorithm is presented in Algorithm 23.

Algorithm 23 Segmentation of blurred images (wsc_3)

function $\nu \leftarrow \text{WSC}_3(\nu^{(0)}, k^{(0)}, z, maxit)$
  $u^{(0)} \leftarrow z$, $\psi^{(0)} \leftarrow \xi_{\boldsymbol{a}}\left(u^{(0)}\right)$, $\omega^{(0)} \leftarrow \xi_{\boldsymbol{b}}\left(k^{(0)}\right)$
  $\varphi_1^{(0)} \leftarrow \mathbf{1}$, $\varphi_2^{(0)} \leftarrow \mathbf{1}$
  for $\ell \leftarrow 1 : maxit$ **do**
    Calculate $c_1^{(\ell)} \leftarrow c_1\left(\psi^{(\ell-1)}, \nu^{(\ell-1)}\right)$, $c_2^{(\ell)} \leftarrow c_2\left(\psi^{(\ell-1)}, \nu^{(\ell-1)}\right)$ using (8.15)
    Update $u^{(\ell)}$ using (8.16)
    Update $\psi^{(\ell)}$ using (8.17) and time marching
    Update $k^{(\ell)}$ using (8.18)
    Update $\omega^{(\ell)}$ using (8.19) and time marching
    Update $\nu^{(\ell)}$ using (8.14) and AOS
    Update $\varphi_1^{(\ell)} \leftarrow \varphi_1^{(\ell-1)} + \gamma_1\left(u^{(\ell)} - \tau_{\boldsymbol{a}}\left(\psi^{(\ell)}\right)\right)$
    Update $\varphi_2^{(\ell)} \leftarrow \varphi_2^{(\ell-1)} + \gamma_2\left(k^{(\ell)} - \tau_{\boldsymbol{b}}\left(\omega^{(\ell)}\right)\right)$
  end for
  $\nu \leftarrow \nu^{(\ell)}$
end function

8.5 An Accelerated Model for the Segmentation of Blurred Images

In order to improve the speed of the model, we consider the problems of segmentation and deblurring separately. We segment the image subject to the condition that the image must be the solution of the deconvolution problem. We thus solve the unconstrained problem

$$\min_{\nu, c_1, c_2} \mu \int_\Omega |\nabla \nu| + \lambda_1 \int_\Omega |z - c_1|^2 \nu \, \mathrm{d}\Omega + \lambda_2 \int_\Omega |z - c_2|^2 (1 - \nu) \, \mathrm{d}\Omega + \sigma \int_\Omega b_\varepsilon H_\epsilon \left(b_\varepsilon\right) \mathrm{d}\Omega$$

where $b_\varepsilon := b_\varepsilon(\nu)$ such that $z(x, y) = u(x, y)$ which should be the solution of Equation (8.1). We introduce a new variable $\varpi(x, y)$ which should be equal to the image at

convergence. We present our joint problem functional as

$$\max_{\varphi_1,\varphi_2,\zeta}\min_{\nu,c_1,c_2,\varpi,u,\psi,k,\omega}\Big\{f_{\mathrm{wsc}_4}(\nu,c_1,c_2,\varpi,u,\psi,k,\omega;\varphi_1,\varphi_2,\zeta)=\mu\int_\Omega|\nabla\nu|$$
$$+\lambda_1\int_\Omega|\varpi-c_1|^2\nu\,\mathrm{d}\Omega+\lambda_2\int_\Omega|\varpi-c_2|^2(1-\nu)\,\mathrm{d}\Omega$$
$$+\sigma\int_\Omega b_\varepsilon(\nu)H_\varepsilon\left(b_\varepsilon(\nu)\right)\mathrm{d}\Omega+\frac{1}{2}\left\|k*u-z\right\|^2_{L^2(\Omega)}+\alpha_1\int_\Omega|\nabla\tau_{\boldsymbol{a}}(\psi)|$$
$$+\frac{\gamma_1}{2}\left\|u-\tau_{\boldsymbol{a}}(\psi)\right\|^2_{L^2(\Omega)}+<\varphi_1,u-\tau_{\boldsymbol{a}}(\psi)>+\frac{\upsilon}{2}\left\|\tau_{\boldsymbol{a}}(\psi)-\varpi\right\|^2_{L^2(\Omega)}+$$
$$<\zeta,\tau_{\boldsymbol{a}}(\psi)-\varpi>+\alpha_2\int_\Omega|\nabla\tau_{\boldsymbol{b}}(\omega)|+\frac{\gamma_2}{2}\left\|k-\tau_{\boldsymbol{b}}(\omega)\right\|^2_{L^2(\Omega)}$$
$$+<\varphi_2,k-\tau_{\boldsymbol{b}}(\omega)>\Big\}. \tag{8.20}$$

In order to solve this model, we derive the partial differential equations defined by the first order optimality conditions. We will take the Euler-Lagrange equations for each argument in turn. Minimising with respect to ν, we obtain the equation derived from $\partial f_{\mathrm{wsc}_4}/\partial\nu$:

$$\mathcal{E}_1(\nu)=\mu\nabla\cdot\left(\frac{\nabla\nu}{|\nabla\nu|}\right)+\lambda_1\int_\Omega|\varpi-c_1|^2\,\mathrm{d}\Omega-\lambda_2\int_\Omega|\varpi-c_2|^2\,\mathrm{d}\Omega$$
$$+\frac{2\sigma(2\nu-1)}{b_\varepsilon(\nu)+1}\left(H_\varepsilon\left(b_\varepsilon(\nu)\right)+\delta_\varepsilon\left(b_\varepsilon(\nu)\right)b_\varepsilon(\nu)\right). \tag{8.21}$$

We find a solution to $\mathcal{E}_1(\nu)=0$ (8.21) using AOS [132, 203]. Letting

$$f:=-\lambda((\varpi-c_1)^2-(\varpi-c_2)^2)--\frac{2\sigma(2\nu-1)}{b(\nu)+1}\left(H_\varepsilon\left(b_\varepsilon(\nu)\right)\delta_\varepsilon\left(b_\varepsilon(\nu)\right)b_\varepsilon(\nu)\right)$$

and denoting $W=|\nabla\nu|^{-1}$, $\nu_t=-\mathcal{E}_1(\nu)$ can be rewritten in the form

$$\frac{\partial\nu}{\partial t}=\mu\Big(\partial_x(W\partial_x\nu)+\partial_y(W\partial_y\nu)\Big)+f.$$

After discretisation, we rewrite in the matrix-vector form ($\hat{\nu}^n=\nu^n+\tau f^n$):

$$\nu^{n+1}=\frac{1}{2}\sum_{\ell=1}^{2}\left(I-2\tau\mu A_\ell(\nu^n)\right)^{-1}\hat{\nu}^n. \tag{8.22}$$

Here, A_ℓ is the diffusion quantity in the ℓ direction ($\ell=1,2$ for x and y directions respectively).

Minimising f_{wsc_4} with respect to u, we obtain the equation

$$k^\dagger*k*u+(\gamma_1+\varphi_1)u=k^\dagger*z+(\gamma_1+\varphi_1)\tau_{\boldsymbol{a}}(\psi) \tag{8.23}$$

where $k^\dagger(x,y)=k(-x,-y)$. It is important to note that after the discretisation of this equation, the term $k^\dagger*k$ along with the operation of convolution defines a symmetric positive definite (spd) matrix. Put briefly, if $\boldsymbol{k}$ and $\boldsymbol{u}$ are the discretised k and u

respectively, then we have

$$\bar{\boldsymbol{k}} \circ \boldsymbol{k} \circ \boldsymbol{u} \;=\; \boldsymbol{H}\boldsymbol{u}$$

for an spd matrix $\boldsymbol{H}$ where $\circ$ denotes the operation of discrete convolution. We can solve this problem using preconditioned conjugate gradient (pcg) method with a preconditioner, following the idea of [195] given by

$$P = (\tilde{k} * \tilde{k} + \chi I)^{\frac{1}{2}}(\chi I + (\gamma_1 + \varphi_1))(\tilde{k} * \tilde{k} + \chi I)^{\frac{1}{2}} \tag{8.24}$$

where $\tilde{k}$ is a circulant approximation to k. Minimising f_{wsc_4} now with respect to ψ, we obtain the equation

$$\mathcal{E}_2(\psi) = \alpha R(\psi) - \tau'_{\boldsymbol{a}}(\psi)(\gamma_1 + \varphi_1)(u - \tau_{\boldsymbol{a}}(\psi)) + \tau'_{\boldsymbol{a}}(\psi)(\upsilon + \zeta)(\tau_{\boldsymbol{a}}(\psi) - \varpi) \tag{8.25}$$

where $R(\psi)$ is the derivative of the regularisation term $\int_\Omega |\nabla \tau_{\boldsymbol{a}}(\psi)|$. We can solve (8.25) using semi-implicit time marching, $\psi_t = -\mathcal{E}_2(\psi)$ by discretising the time step as above.

Now, minimising f_{wsc_4} with respect to ϖ, we obtain

$$\mathcal{E}_3(\varpi) = 2\lambda_1(\varpi - c_1)\nu + 2\lambda_2(\varpi - c_2)(1 - \nu) + u(\varpi - \tau_{\boldsymbol{a}}(\psi)) + \zeta^\dagger(\varpi - \tau_{\boldsymbol{a}}(\psi)).$$

Note that we can solve the sub-problem $\mathcal{E}_3(\varpi) = 0$ directly by giving the solution to

$$\left(2\lambda_1\nu + 2\lambda_2(1 - \nu) + u + \zeta^\dagger\right)\varpi = 2\lambda_1 c_1\nu + 2\lambda_2 c_2(1 - \nu) + \left(u + \zeta^\dagger\right)\tau_{\boldsymbol{a}}(\psi). \tag{8.26}$$

Finally, minimising f_{wsc_4} with respect to c_1 and c_2, we obtain equations which can be evaluated directly:

$$c_1(\varpi, \nu) = \tau_{\boldsymbol{a}}\left(\frac{\int_\Omega \varpi\nu \,\mathrm{d}\Omega}{\int_\Omega \nu \,\mathrm{d}\Omega}\right), \qquad c_2(\varpi, \nu) = \tau_{\boldsymbol{a}}\left(\frac{\int_\Omega \varpi(1 - \nu) \,\mathrm{d}\Omega}{\int_\Omega 1 - \nu \,\mathrm{d}\Omega}\right). \tag{8.27}$$

Finally, minimising with respect to k and ω, we obtain equations (8.18) and (8.19) respectively.

In order to solve the model (8.20), we make an initial estimate of the image, which we allow to be given by the received data z since this is the closest approximation to the true data that we have. Alternatively, if we know or can make an estimate of the point spread function, we may solve a Tikhonov model [193] and attempt to use this as the initial estimate based on visual judgement. We then calculate the initial estimate of ψ as the inverse transform of the initial estimate of the image. Similarly, in the blind case, we make an initial estimate of the point spread function based on visual observation and compute its inverse transform function. We next make an initial estimate of the contour, obtaining the initial estimate of ν. Using these and (8.27), we make the initial estimates of c_1 and c_2. We then proceed to solve the model (8.20), alternately minimising with respect to the arguments. The final segmentation is then given by the contour Γ_p derived from the final function ν. We present this algorithm in Algorithm 24 below.

Algorithm 24 Segmentation of Blurred Images (wsc_4)

function $\nu \leftarrow \text{WSC}_4(\nu^{(0)}, k^{(0)}, z, maxit)$
  $u^{(0)} \leftarrow z$, $\varpi^{(0)} \leftarrow u^{(0)}$, $\psi^{(0)} \leftarrow \xi_{\boldsymbol{a}}\left(u^{(0)}\right)$, $\omega^{(0)} \leftarrow \xi_{\boldsymbol{b}}\left(k^{(0)}\right)$
  $\varphi_1^{(0)} \leftarrow \mathbf{1}$, $\varphi_2^{(0)} \leftarrow \mathbf{1}$, $\zeta^{(0)} \leftarrow \mathbf{1}$
  for $\ell \leftarrow 1 : maxit$ **do**
    Calculate $c_1^{(\ell)} \leftarrow c_1\left(\varpi^{(\ell-1)}, \nu^{(\ell-1)}\right)$, $c_2^{(\ell)} \leftarrow c_2\left(\varpi^{(\ell-1)}, \nu^{(\ell-1)}\right)$ using (8.27)
    Update $u^{(\ell)}$ using (8.23)
    Update $\psi^{(\ell)}$ using (8.25) and time marching
    Update $\nu^{(\ell)}$ using (8.21) and AOS
    Update $\varpi^{(\ell)}$ using (8.26)
    Update $\varphi_1^{(\ell)} \leftarrow \varphi_1^{(\ell-1)} + \gamma_1\left(u^{(\ell)} - \tau_{\boldsymbol{a}}\left(\psi^{(\ell)}\right)\right)$
    Update $\varphi_2^{(\ell)} \leftarrow \varphi_2^{(\ell-1)} + \gamma_2\left(u^{(\ell)} - \tau_{\boldsymbol{b}}\left(\omega^{(\ell)}\right)\right)$
    Update $\zeta^{(\ell)} \leftarrow \zeta^{(\ell-1)} + \upsilon\left(\tau_{\boldsymbol{a}}\left(\psi^{(\ell)}\right) - w^{(\ell)}\right)$
  end for
  $\nu \leftarrow \nu^{(\ell)}$
end function

8.6 Experimental Results

In this section, we demonstrate that attempting to segment a blurred image with Chan-Vese is sufficient to obtain a close result if the degradation is not strong but as the amount of corruption increases, Chan-Vese is not capable of obtaining a good result because it was not designed with blur degradation in mind. Meanwhile, the work of Bar, Sochen and Kiriyati [10] is capable of segmenting blurred images whether the corruption is small but begins to struggle to obtain good quality results in the presence of significant blur degradation or noise. We expect that wsc_3 will be capable of obtaining a good quality result with the possibility of slow convergence while wsc_4 will converge faster to a similar, if slightly lower, quality.

We present results of segmenting the following images (See Figure 8.2) with the addition of varying levels of Gaussian blur and noise: Im_1: Text (Fig. 8.2a), Im_2: Cells medical (Fig. 8.2b), Im_3: Box-Triangle (Fig. 8.2c), Im_4: QR Code (Fig. 8.2d), Im_5: Fingerprint (Fig. 8.2e), Im_6: Tree (Fig. 8.2f).

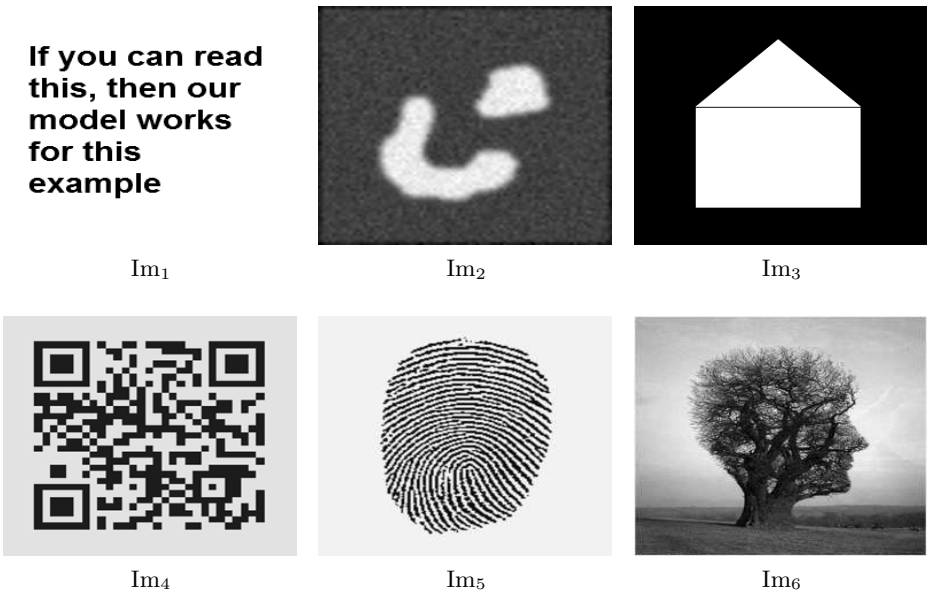
We denote by Γ_{Mod_1} and ν_{Mod_1} respectively the contour obtained and the segmented area obtained by solving Mod_1. The notation is similar for the other models.

Experiments were carried out using Matlab R2013a on a HPE-595uk with an Intel(R) Core(TM) i7-2600 processor and 16GB RAM.

8.6.1 Models

In order to compare our results with competing and other relevant models, we define the following models to be tested in this section:

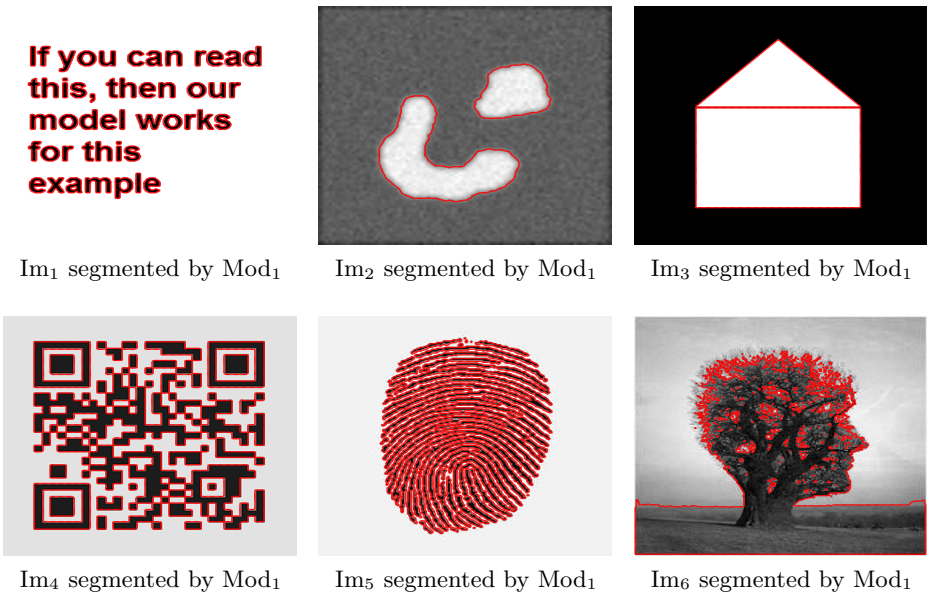
Mod_1: The Chan-Vese model [53] given by solving the minimisation problem (3.57).



Im_1 Im_2 Im_3

Im_4 Im_5 Im_6

Figure 8.2: Images used for test examples.



Im_1 segmented by Mod_1 Im_2 segmented by Mod_1 Im_3 segmented by Mod_1

Im_4 segmented by Mod_1 Im_5 segmented by Mod_1 Im_6 segmented by Mod_1

Figure 8.3: Segmentation of Images Im_1–Im_6 using model Mod_1.

Mod_2: The two-stage model given by deblurring using (3.25) and segmenting using (3.57).

Mod_3: The two-stage model given by deblurring using (3.35) and segmenting using (3.57).

Mod_4: The segmentation of blurred images model [10].

New_8^1: The two-stage model given by Algorithm 21.

New_8^2: The two-stage model given by Algorithm 22.

New_8^3: The model given by solving (8.13) using Algorithm 23.

New_8^4: The model given by solving (8.20) using Algorithm 24.

8.6.2 Measuring Error

In order to make a numerical evaluation of our model, we require some ground truth. For the artificial images Im_1 and Im_3, we already know the contour and consider this to be the ground truth segmentation. For the remaining images, we estimate the true contour by assuming that the segmentation of the true (uncorrupted) image is correct (see Figure 8.3) and we consider methods of measuring the accuracy of the final contour Γ, that is how close it is to the segmentation by Chan-Vese of the true image given by Γ^{true}. To this end, we compare the final contours and indicator functions. We also compare the length of the contours given by the segmentation by each model. It would be useful to measure the distance of Γ from Γ^{true} but since we cannot identify corresponding points for comparison, we measure the distance to the nearest point, that is for each point $\Gamma_{i,j}$ on the contour Γ, we locate the point $\Gamma^{\text{true}}_{k,l}$ such that $r = (i-k)^2 + (j-l)^2$ is minimised. The distance is then given by the square root of r. We compare results using the following error measures, each of which tend to zero as the segmentation of the blurred image tends towards the segmentation of the true image. Throughout, we denote the true, received and restored image u as u^{true}, u^{rec} and u^{rest} respectively, with similar notation for the remaining functions. To distinguish models, we also denote by $\Gamma_{\text{New}_8^3}$ the contour obtained by solving model New_8^3 and adopt similar notation for the remaining models and functions.

- L2 area-based difference gives the L^2-norm of the difference in segmented images. It measures the closeness of the final indicator functions:

$$Er_1 = \left\|\nu^{\text{rest}} - \nu^{\text{true}}\right\|_2^2$$

  where ν^{rest} corresponds to the function achieved from solving the model.

- Contour difference gives the L^2-norm of the difference between final contours:

$$Er_2 = \left\|\Gamma^{\text{rest}} - \Gamma^{\text{true}}\right\|_2^2.$$

- Contour length difference. Chan-Vese constrain the length l_Γ of the contour Γ using total variation $l_\Gamma = \int_\Omega |\nabla H(\psi)|$ where the contour Γ is represented by the zero level-set of the Lipshchitz function ϕ and $H(x)$ is the Heaviside function. We present our contour length distance measure as the normalised absolute value of the difference in contour length:

$$Er_3 = \frac{\left|\int_\Omega |\nabla \nu^{\text{rest}}| - \int_\Omega |\nabla \nu^{\text{true}}|\right|}{\left|\int_\Omega |\nabla \nu^{\text{rec}}| - \int_\Omega |\nabla \nu^{\text{true}}|\right|}.$$

- Mean closest point measure. Here, we measure the distance of the closest point of Γ^{true} to each point of Γ^{rec}. Since this is a point-wise calculation and different approximations of Γ may contain different numbers of points, we give the mean value:

$$Er_4 = \frac{1}{n^\iota} \sum_{\substack{i,j \\ \iota_{i,j} \neq 0}} r^\iota(i,j,i_\iota,j_\iota), \qquad (i_\iota, j_\iota) = \min_{i_\iota, j_\iota} r^\iota(i,j,i_\iota,j_\iota),$$

where

$$r^\iota(i,j,i_\iota,j_\iota) = \sqrt{(i-i_\iota)^2 + (j-j_\iota)^2},$$

$$n^\iota = \#\left\{x \in \iota | x \neq 0\right\}, \qquad \iota_{i,j} = \begin{cases} 1 & \text{if } \Gamma^{\text{rec}} \geq l \\ 0 & \text{otherwise.} \end{cases}$$

- Letting the set of points which are considered to be inside in the contour be given by

$$S^{\text{rest}} = \left\{(x,y) \in \Omega | \nu^{\text{rest}}(x,y) > 10^{-1}\right\},$$

$$S^{\text{true}} = \left\{(x,y) \in \Omega | \nu^{\text{true}}(x,y) > 10^{-1}\right\},$$

we define Er_5 (Tannimoto Coefficient) [72, 108, 65] and Er_6 (Dice Similarity Coefficient) [67, 65] as

$$Er_5 = \frac{\mathcal{N}\left(S^{\text{rest}} \cap S^{\text{true}}\right)}{\mathcal{N}\left(S^{\text{rest}} \cup S^{\text{true}}\right)} \quad \text{and} \quad Er_6 = \frac{2\mathcal{N}\left(S^{\text{rest}} \cap S^{\text{true}}\right)}{\mathcal{N}\left(S^{\text{rest}}\right) + \mathcal{N}\left(S^{\text{true}}\right)}$$

respectively where $\mathcal{N}(S)$ denotes the number of elements in the set S. It is clear that as the restored segmentation tends towards the true segmentations, both error values tend toward one.

8.6.3 Result Sets

We group our experimental results by the following result sets:

RS_1: Result set 1 consists of images corrupted by blur and Poisson noise. We illustrate the performance of Mod_1. We see in Tables 8.1 and 8.8 that while Mod_1 can give a reasonable result, it can be improved by the constrained model New_8^1.

RS_2: Result set 2 consists of images corrupted by blur and Poisson noise. We demonstrate in Table 8.2 that model New_8^2 can give an improvement over Mod_3.

RS_3: Result set 3 consists of images corrupted by small amounts of blur. We demonstrate in Tables 8.3–8.4 and Figures 8.4–8.6 that model Mod_1 is sufficient to obtain a close result but Mod_4 can give an improvement. We also demonstrate that the new models are capable of working with these examples and offer further enhancement.

RS_4: Result set 4 consists of images corrupted by larger amounts of blur. We demonstrate in Tables 8.5–8.6 and Figure 8.7 that Mod_1 is insufficient for these examples, Mod_4 can work reasonably well but New_8^3 offers further improvement.

RS_5: Result set 5 consists of images corrupted by blur and noise. We demonstrate in Table 8.7 and Figure 8.8 that Mod_1 fails to get good results but New_8^3 continues to perform well.

RS_6: Result set 6 demonstrates the ability of New_8^4 and compares the performance of this model with New_8^3. We can see in in Tables 8.3–8.7 and Figure 8.9 that it is generally the case that New_8^4 is faster than New_8^3 while New_8^3 obtains better results.

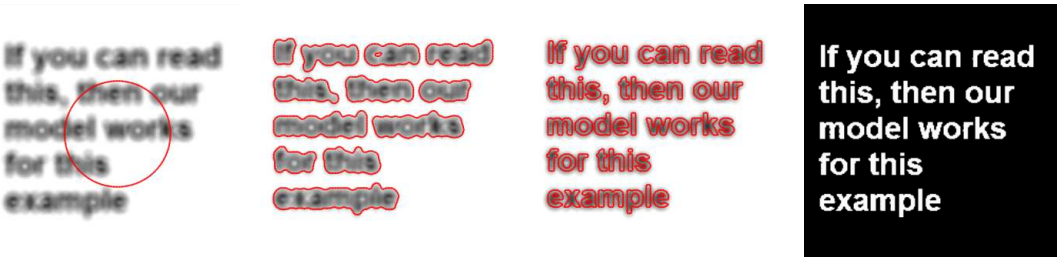
RS_7: Result set 7 compares the results from New_8^3–New_8^4 with the results from New_8^1. We can see in in Table 8.8 that there is little advantage in considering the problems of deblurring and segmentation separately. In one case, the result is improved by the two-stage model but in the remaining cases, the joint models achieve better results.

		Small Gaussian Blur, $\sigma = 9$			Large Gaussian Blur, $\sigma = 19$		
Image	Model	Er_1	Er_2	Er_5	Er_1	Er_2	Er_5
Im_1	Initial	101.86	86.19	0.3703	141.48	82.47	0.2227
	Mod_2	**81.59**	**71.21**	**0.4840**	**44.82**	**65.81**	**0.8532**
Im_2	Initial	32.27	36.30	0.8894	63.59	37.38	0.6733
	Mod_2	**18.99**	**23.73**	**0.9702**	**14.73**	**28.28**	**0.8942**
Im_3	Initial	16.79	23.81	0.9864	32.10	36.70	0.9489
	Mod_2	**1.62**	**0**	**1**	**1.60**	**0**	**1**
Im_4	Initial	81.87	100.09	0.7434	142.16	91.71	0.4484
	Mod_2	**21.69**	**30.98**	**0.8942**	**48.71**	**68.60**	**0.8824**
Im_5	Initial	120.87	109.63	0.4857	134.30	109.44	0.4169
	Mod_2	**44.96**	**66.06**	**0.8658**	**43.54**	**62.15**	**0.8735**
Im_6	Initial	69.01	75.80	0.8488	84.28	75.87	0.7856
	Mod_2	**46.22**	**67.20**	**0.8808**	**33.62**	**48.11**	**0.9582**

Table 8.1: Result set 1. Error values for Im_1–Im_6 corrupted by Gaussian blur and segmented by Mod_1.

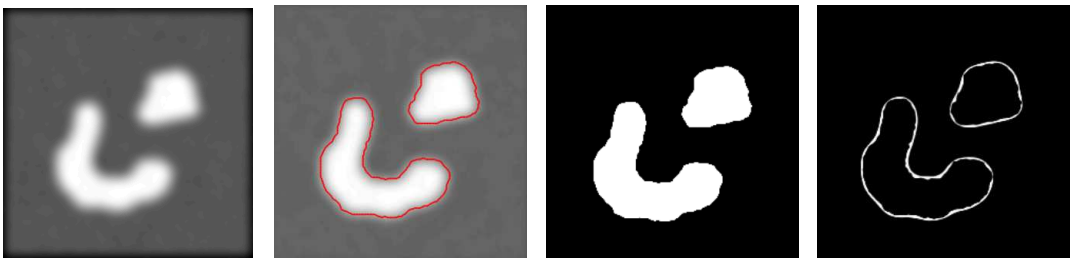
Image	Model	Small Gaussian Blur, $\sigma = 9$			Large Gaussian Blur, $\sigma = 19$		
		Er_1	Er_2	Er_5	Er_1	Er_2	Er_5
Im_1	Initial	101.86	86.19	0.3703	141.48	82.47	0.2227
	Mod_3	72.97	77.32	0.5350	96.44	78.63	0.3908
	New_8^2	**67.52**	**75.39**	**0.5731**	**90.31**	**74.70**	**0.4268**
Im_2	Initial	32.27	36.30	0.8894	63.59	37.38	0.6733
	Mod_3	**11.73**	18.63	0.9820	25.72	33.06	0.9236
	New_8^2	11.74	**18.47**	**0.9823**	**25.50**	**32.94**	**0.9248**
Im_3	Initial	16.80	23.81	0.9864	32.11	36.70	0.9489
	Mod_3	12.29	**16.94**	**0.9930**	12.49	17.49	0.9925
	New_8^2	**11.97**	**16.94**	**0.9930**	**12.28**	**17.29**	**0.9927**
Im_4	Initial	81.87	100.09	0.7434	142.16	91.72	0.4484
	Mod_3	57.22	81.51	0.8574	110.57	89.48	0.6121
	New_8^2	**55.51**	**79.34**	**0.8647**	**110.51**	**89.40**	**0.6124**
Im_5	Initial	120.87	109.63	0.4857	134.30	109.44	0.4169
	Mod_3	102.98	**103.05**	0.5673	104.25	103.54	0.5607
	New_8^2	**102.95**	**103.05**	**0.5675**	**104.21**	**103.51**	**0.5610**
Im_6	Initial	69.01	75.80	0.8488	84.29	75.88	0.7856
	Mod_3	**50.96**	**69.64**	**0.9089**	**57.27**	70.41	**0.8897**
	New_8^2	50.97	**69.64**	**0.9089**	57.27	**70.38**	**0.8897**

Table 8.2: Result set 2. Error values for Im_1–Im_6 corrupted by Gaussian blur and segmented by Mod_3 and New_8^2. The competition is close for most examples, but overall New_8^2 outperforms Mod_3.



(a) Initial Contour (b) Segmented Γ_{Mod_1} (c) Segmented $\Gamma_{\text{New}_8^3}$ (d) Segmented $\nu_{\text{New}_8^3}$

Figure 8.4: Illustration of the performance of the Mod_1 for Im_1 corrupted by Gaussian blur: a) initial contour, b) segmentation given by Mod_1, c,d) segmentation given by New_8^3. Mod_1 gives a rough segmentation while the spaces between the letters which are hidden by the blur are successfully segmented using New_8^3.



(a) Received image (b) Segmented $\Gamma_{\text{New}_8^3}$ (c) Segmented $\nu_{\text{New}_8^3}$ (d) $\nu_{\text{Mod}_1} - \nu_{\text{New}_8^3}$

Figure 8.5: Illustration of the performance of the New_8^3 for Im_2 corrupted by Gaussian blur: a) received data, b,c) segmentation using New_8^3, d) the difference between the segmentation using New_8^3 and using Mod_1. The segmentation is closer to the true edge using New_8^3 while Mod_1 also captures the blurred edge.

Model	Er_1	Er_2	Er_3	Er_4	Er_5	Er_6	cpu
			Im_1 corrupted by small Gaussian blur				
Initial	101.1621	64.3428	1	27.7407	0.1326	0.2341	
Mod_1	101.8554	86.1916	0.0588	3.9898	0.3703	0.5405	**316.85**
Mod_4	94.9352	85.9884	0.0482	3.7122	0.4045	0.5761	1361.50
New_8^3	**15.3386**	**23.1948**	0.0080	**0.0667**	**0.9595**	**0.9793**	559.09
New_8^4	16.5861	24.5357	**0.0063**	0.0811	0.9548	0.9769	442.53
			Im_2 corrupted by small Gaussian blur				
Initial	104.3123	32.0312	1	19.7068	0.2289	0.3725	
Mod_1	32.2705	36.3043	0.0130	1.7335	0.8895	0.9415	526.55
Mod_4	23.0196	31.9374	0.0130	0.8778	0.9412	0.9697	861.03
New_8^3	**5.8182**	**11.1355**	**0.0065**	**0.0912**	**0.9936**	**0.9968**	546.49
New_8^4	6.9165	12.3288	**0.0065**	0.1148	0.9922	0.9961	**426.65**
			Im_3 corrupted by small Gaussian blur				
Initial	109.6528	34.8425	1	25.3548	0.4218	0.5934	
Mod_1	16.7986	23.8118	0.5763	8.9809	0.9863	0.9931	319.26
Mod_4	13.6400	19.4422	0.5663	8.6980	0.9909	0.9954	634.46
New_8^3	**1.0923**	**0**	**0**	**0**	**1**	**1**	550.14
New_8^4	1.5249	**0**	**0**	**0**	**1**	**1**	**227.54**
			Im_4 corrupted by small Gaussian blur				
Initial	138.1870	77.4855	1	31.6798	0.1920	0.3221	
Mod_1	81.8720	100.0950	0.0711	1.5198	0.7434	0.8528	527.27
Mod_4	65.7515	84.3386	0.1398	0.9650	0.8178	0.8997	941.75
New_8^3	**14.3647**	**20.2978**	0.0118	**0.0407**	**0.9892**	**0.9946**	586.11
New_8^4	17.6317	25.1595	**0.0105**	0.0499	0.9835	0.9917	**331.33**

Table 8.3: Result sets 3, 6. Error values and cpu times for images Im_1–Im_4 corrupted by small Gaussian blur. Error values are improved with New_8^3 and New_8^4. New_8^3 achieves better error values with the exception of contour length which is closer with or identical to the result from New_8^4 while New_8^4 achieves the lowest cpu time. For Im_1, the cpu time is lower for Mod_1, but the error values are considerably deteriorated.

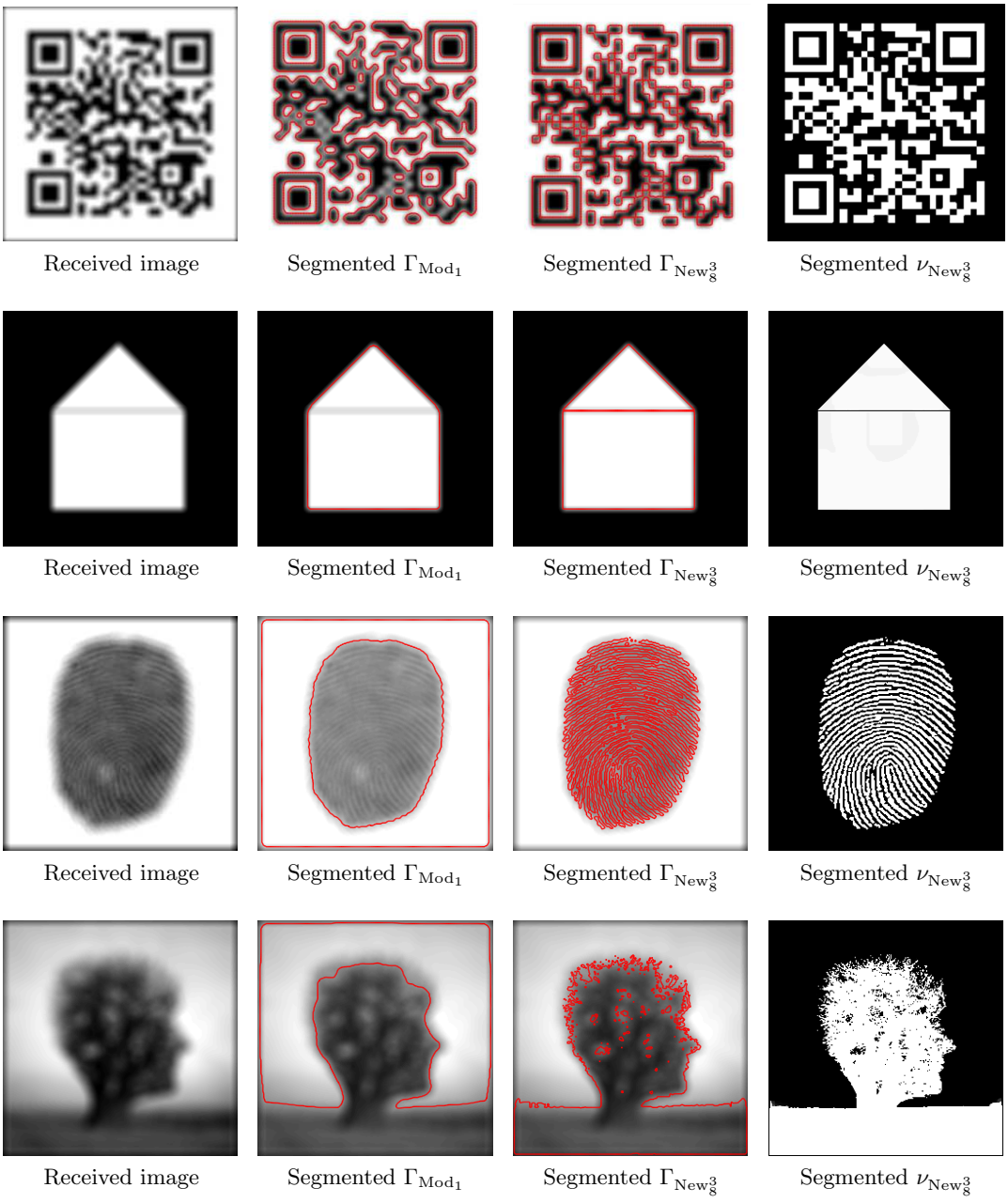


Figure 8.6: Result set 3. Illustration of the performance of New_8^3 for (top-bottom) Im_4, Im_3, Im_5 and Im_6 corrupted by Gaussian blur. The edges hidden by blur are successfully segmented by New_8^3 which cannot be segmented by Mod_1.

Model	Er_1	Er_2	Er_3	Er_4	Er_5	Er_6	cpu
			Im_5 corrupted by small Gaussian blur				
Initial	109.0745	102.4695	1	20.7918	0.291	0.4509	
Mod_1	120.8688	109.6266	0.8239	27.0090	0.4857	0.6539	**523.76**
Mod_4	113.2628	110.0091	0.8235	27.4588	0.5204	0.6846	834.57
New_8^3	**28.4287**	**45.8694**	**0.0133**	**0.0958**	**0.9280**	**0.9626**	567.86
New_8^4	38.0463	54.5894	0.0145	0.1531	0.8993	0.947	542.74
			Im_6 corrupted by small Gaussian blur				
Initial	145.9686	68.5128	1	27.6162	0.2748	0.4311	
Mod_1	69.0106	75.8024	0.6406	11.9931	0.8488	0.9182	**524.60**
Mod_4	61.1291	75.0866	0.6689	12.0802	0.8774	0.9347	835.85
New_8^3	**32.4118**	**48.6107**	0.3949	**0.1719**	**0.959**	**0.9791**	562.10
New_8^4	35.3429	50.4876	**0.3724**	0.9943	0.9532	0.9761	536.60
			'Circles' image corrupted by small Gaussian blur				
Initial	65.2066	28.2843	1	10.4102	0.6921	0.818	
Mod_1	37.5223	30.0167	0.3649	3.5455	0.8902	0.9419	423.12
Mod_4	33.9973	29.8998	0.2703	2.8027	0.9111	0.9535	730.72
New_8^3	18.6340	**26.9815**	**0.0811**	0.8284	**0.9723**	**0.9860**	329.91
New_8^4	**18.5900**	**26.9815**	**0.0811**	**0.8261**	**0.9723**	**0.9860**	**319.83**
			'Knee' image corrupted by small Gaussian blur				
Initial	120.5369	60.6960	1	30.8333	0.3241	0.4895	
Mod_1	78.0728	63.5610	0.8166	14.1297	0.7642	0.8663	**538.47**
Mod_4	72.5519	64.9615	0.6942	7.1083	0.7848	0.8794	1166.30
New_8^3	34.1267	47.1063	0.2459	**0.3409**	0.9305	0.9640	560.62
New_8^4	**32.3627**	**45.6508**	**0.1054**	0.5914	**0.9423**	**0.9703**	542.48

Table 8.4: Result sets $3, 6$. Error values and cpu times for Im_5, Im_6, 'Circles' and 'Knee' images corrupted by small Gaussian blur. In all cases, New_8^3 and New_8^4 achieve improved results. The contour length is typically closer with New_8^4 and while for most examples the cpu time is lower for Mod_1, it is closely followed by New_8^4 which gives considerably better results.

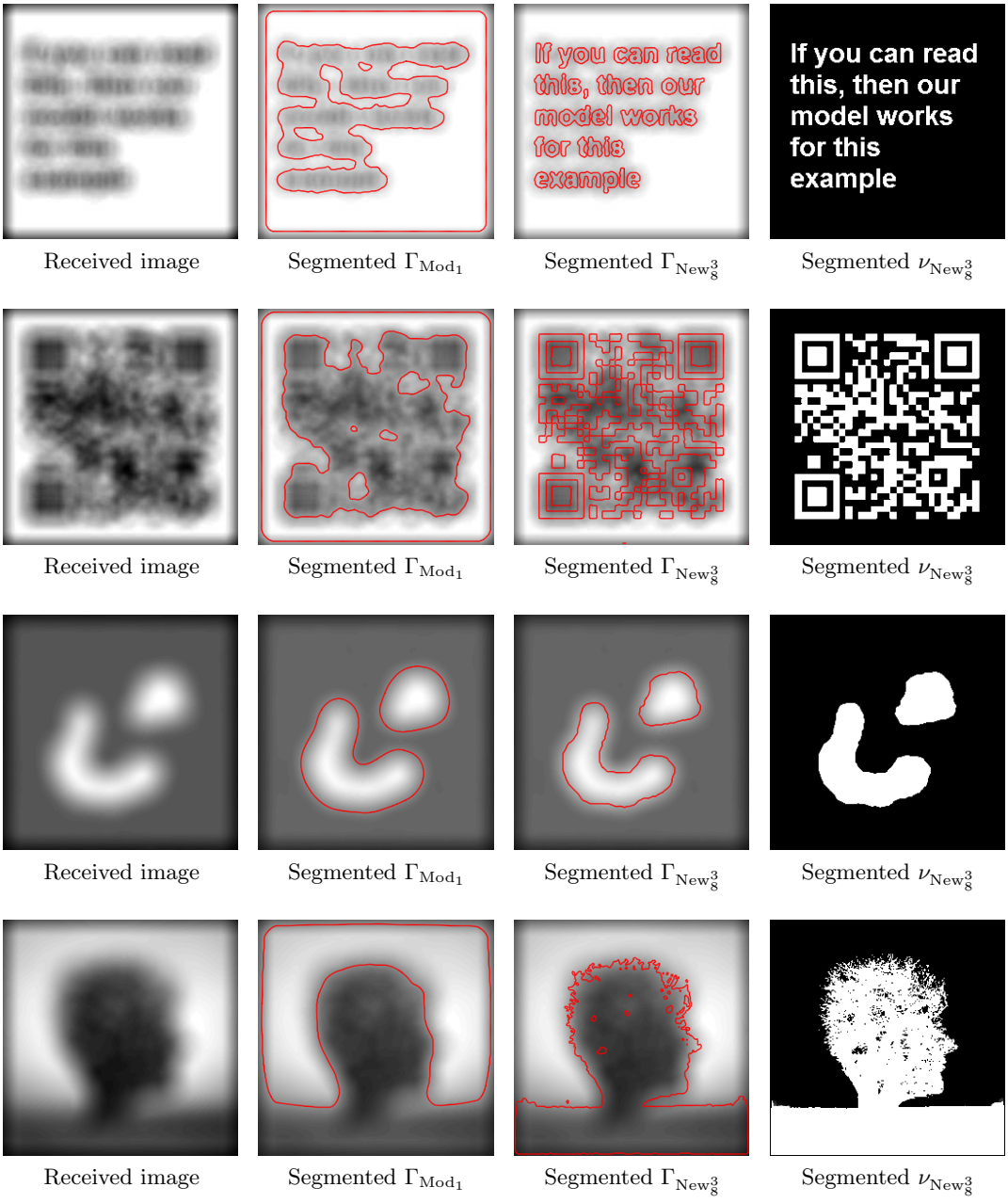


Figure 8.7: Result set 4. Illustration of the performance of the New_8^3 for (top-bottom) Im_1, Im_4, Im_2 and Im_6 corrupted by strong Gaussian blur. New_8^3 is capable of segmenting edges in these challenging cases which cannot be segmented by Mod_1.

Model	Er_1	Er_2	Er_3	Er_4	Er_5	Er_6	cpu
			Im_1 corrupted by large Gaussian blur				
Initial	101.1621	64.3428	1	27.7407	0.1326	0.2341	
Mod_1	141.4771	82.4682	0.2275	7.5976	0.2227	0.3643	**520.04**
Mod_4	151.0805	80.3990	0.3494	12.0389	0.2085	0.3451	1478.92
New_8^3	**35.6765**	**54.8270**	**0.0516**	**0.3855**	**0.8021**	**0.8902**	557.49
New_8^4	44.9299	62.9762	0.0631	0.6174	0.7484	0.8561	548.62
			Im_2 corrupted by large Gaussian blur				
Initial	104.3123	32.0312	1	19.7068	0.2289	0.3725	
Mod_1	63.5872	37.3765	0.1786	6.9414	0.6733	0.8047	522.49
Mod_4	47.1706	36.8782	0.0649	3.8043	0.7888	0.882	949.50
New_8^3	**6.7929**	**13.8564**	0.0195	**0.1398**	**0.9902**	**0.9951**	566.25
New_8^4	10.1678	16.3095	**0.0065**	0.1997	0.9863	0.9931	**433.78**
			Im_3 corrupted by large Gaussian blur				
Initial	109.6528	34.8425	1	25.3548	0.4218	0.5934	
Mod_1	32.1049	36.7015	0.5803	10.1626	0.9488	0.9737	417.96
Mod_4	28.3128	36.7560	**0.5683**	9.7345	0.9613	0.9803	729.99
New_8^3	**11.9585**	**16.8819**	0.5723	**0**	**0.9931**	**0.9965**	552.04
New_8^4	12.0840	16.9706	0.5743	8.6278	0.9930	**0.9965**	**232.10**
			Im_4 corrupted by large Gaussian blur				
Initial	138.1870	77.4855	1	31.6798	0.1920	0.3221	
Mod_1	142.1617	91.7170	0.5268	12.2782	0.4484	0.6192	524.83
Mod_4	140.1087	91.2524	0.5406	14.2381	0.4928	0.6602	1477.41
New_8^3	**25.0290**	**38.5357**	**0.0031**	0.1943	**0.9649**	**0.9821**	588.69
New_8^4	30.4164	43.5890	0.0126	**0.1592**	0.9547	0.9768	**443.53**

Table 8.5: Result sets 4, 6. Error values and cpu times for images Im_1–Im_4 corrupted by strong Gaussian blur. In all cases, New_8^3 and New_8^4 achieve improved results although the contour length of the results from Mod_4 is better for Im_3. For most cases, the cpu time is lower for New_8^4 with the exception of Im_1 which has slightly lower cpu time for Mod_1 with deteriorated results.

Model	Er_1	Er_2	Er_3	Er_4	Er_5	Er_6	cpu
			Im_5 corrupted by large Gaussian blur				
Initial	109.0745	102.4695	1	20.7918	0.2910	0.4509	
Mod_1	134.2968	109.4395	0.8313	25.8610	0.4169	0.5885	522.67
Mod_4	126.1879	110.5351	0.8273	27.9938	0.4616	0.6317	937.21
New_8^3	**74.0995**	**91.3947**	0.5078	**0.3076**	0.6852	0.8132	564.26
New_8^4	74.7745	92.9785	**0.3420**	1.1634	**0.7053**	**0.8272**	**334.29**
			Im_6 corrupted by large Gaussian blur				
Initial	145.9686	68.5128	1	27.6162	0.2748	0.4311	
Mod_1	84.2870	75.8749	0.6721	13.0561	0.7856	0.8799	**520.25**
Mod_4	76.3585	76.0855	0.6658	13.4703	0.8216	0.9021	939.89
New_8^3	**40.1934**	**58.1979**	0.6359	**0.2931**	**0.9412**	**0.9697**	573.62
New_8^4	42.5394	60.4401	**0.6315**	2.4799	0.9354	0.9666	538.69
			'Circles' corrupted by large Gaussian blur				
Initial	65.2066	28.2843	1	10.4102	0.6921	0.818	
Mod_1	114.8976	32.7567	2.6892	28.8661	0.4699	0.6393	525.44
Mod_4	124.1155	33.0757	2.9730	32.2090	0.4381	0.6093	1364.14
New_8^3	26.2901	29.8161	0.2027	2.1064	0.9317	0.9646	542.70
New_8^4	**24.8092**	**29.7489**	**0.1486**	**1.5114**	**0.9493**	**0.974**	**430.17**
			'Knee' corrupted by large Gaussian blur				
Initial	120.5369	60.6960	1	30.8333	0.3241	0.4895	
Mod_1	89.1061	63.1664	0.8808	19.9400	0.7001	0.8236	**521.73**
Mod_4	87.8371	63.1110	0.8543	18.3418	0.7171	0.8352	1148.25
New_8^3	47.2881	56.3915	0.4964	**0.5754**	0.8854	0.9392	548.02
New_8^4	**43.2880**	**55.5338**	**0.3471**	1.6406	**0.9016**	**0.9483**	538.61

Table 8.6: Result sets $4, 6$. Error values and cpu times for Im_5, Im_6, 'Circles' and 'Knee' images corrupted by strong Gaussian blur. In all cases, New_8^3 and New_8^4 achieve improved results and competition is close between New_8^3 and New_8^4. Cpu time is lower for New_8^4 in two cases,. It is lower for Mod_1 is two other cases followed by New_8^4 which achieved considerably better results.

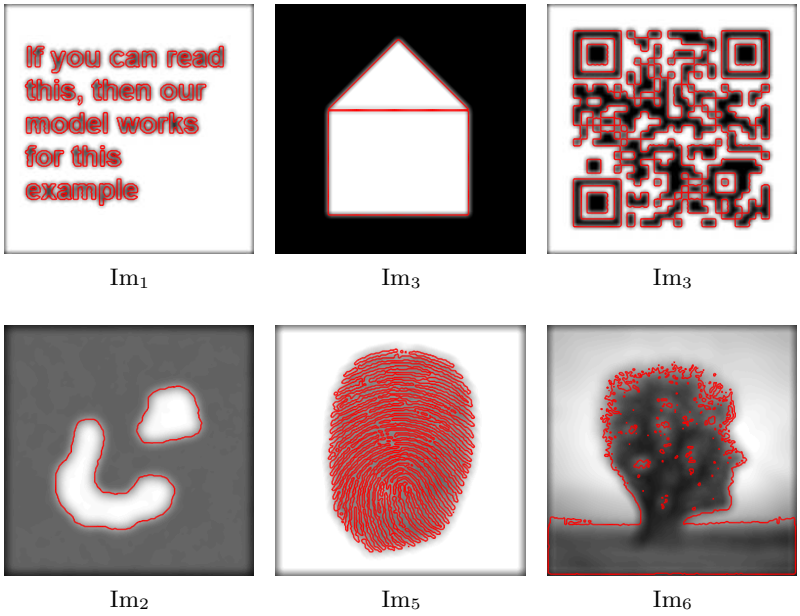
Figure 8.8: Result set 5. Illustration of the performance of the New_8^3 for (top-bottom) Im_1, Im_3, Im_4 and Im_5 corrupted by Gaussian blur and noise. The edges hidden by blur are successfully segmented by New_8^3 which cannot be segmented by Mod_1.

Model	Er_1	Er_2	Er_3	Er_4	Er_5	Er_6	cpu
Im_1 corrupted by small Gaussian blur and noise							
Initial	101.1621	64.3428	1	27.7407	0.1326	0.2341	
Mod_1	101.8449	86.2612	0.0554	3.9891	0.3699	0.5400	**318.69**
Mod_4	94.9266	86.1162	0.0459	3.7143	0.4043	0.5758	1898.59
New_8^3	**13.8647**	**19.8494**	0.0086	0.0547	**0.9669**	**0.9832**	606.52
New_8^4	14.8951	20.9523	**0.0060**	**0.0545**	0.9637	0.9815	452.03
Im_3 corrupted by small Gaussian blur and noise							
Initial	109.6528	34.8425	1	25.3548	0.4218	0.5934	
Mod_1	16.8566	23.9374	0.5763	8.9764	0.9862	0.9931	316.93
Mod_4	14.3153	21.5407	0.5502	8.7471	0.989	0.9945	633.33
New_8^3	**1.7542**	**0**	**0**	**0**	**1**	**1**	511.56
New_8^4	2.1775	**0**	**0**	**0**	**1**	**1**	**242.29**
Im_4 corrupted by small Gaussian blur and noise							
Initial	138.1870	77.4855	1	31.6798	0.1920	0.3221	
Mod_1	81.9065	99.9950	**0.0737**	1.5194	0.7433	0.8527	529.84
Mod_4	65.7144	84.3505	0.1394	0.9616	0.8180	0.8999	957.69
New_8^3	26.8356	36.2491	0.1193	0.3791	0.9639	0.9816	639.88
New_8^4	**23.8973**	**32.1403**	0.0767	**0.0535**	**0.9712**	**0.9854**	**341.99**
Im_5 corrupted by small Gaussian blur and noise							
Initial	109.0745	102.4695	1	20.7918	0.2910	0.4509	
Mod_1	120.8587	109.6403	0.8236	26.9932	0.4857	0.6539	**546.64**
Mod_4	113.2456	109.9818	0.8235	27.4606	0.5205	0.6846	876.65
New_8^3	**41.6099**	**62.1289**	**0.0304**	0.2897	**0.8682**	**0.9295**	599.82
New_8^4	45.3174	63.9531	0.0890	**0.1538**	0.8607	0.9251	575.36

Table 8.7: Result sets 5, 6. Error values and cpu times for Im_1, Im_3–Im_5 corrupted by Gaussian blur and noise. In all cases, New_8^3 and New_8^4 achieve improved results with the exception that the contour length is of the result for Im_4 was closer to the true contour with Mod_1. Cpu time is lower for New_8^4 in two cases. In the remaining cases, it is lower for Mod_1 and closely followed by New_8^4 which achieved significantly improved results.

Image	Initial	New_8^1	New_8^3	New_8^4
Im_1	101.1621	**12.9300**	15.3386	16.5861
Im_2	104.3123	7.2681	**5.8182**	6.9165
Im_3	109.6528	1.5731	**1.0923**	1.5249
Im_4	138.1870	17.9387	**14.3647**	17.6317

Table 8.8: Result set 7. Error values given by Er_1 for Im_1–Im_4 corrupted by Gaussian blur and segmented by New_8^1, New_8^3 and New_8^4. For Im_1, New_8^1 outperforms the other models but in the remaining cases New_8^3 and New_8^4 obtain improved results.



Im_1 Im_3 Im_3

Im_2 Im_5 Im_6

Figure 8.9: Result set 6. Images corrupted by Gaussian blur segmented using New_8^4.

8.7 Conclusion

We have presented a new model for the effective segmentation of blurred images and presented results demonstrating its ability to capture edges which are blurred and difficult to segment closely as well as edges that are hidden by blur. We have also presented an accelerated model which is also capable of achieving good results with similar examples. This model can be further extended to the semi-blind case where some information about the blur function may be assumed to be known (see Chapter 7 and [2, 11, 135, 154, 160, 172]), and to work with multi-channel images [184, 185, 112]. This model could also be extended to selective segmentation and vessel segmentation techniques among others.

Chapter 9

Conclusions and Future Work

In this final chapter, we present the conclusions of this thesis in §9.1 and some ideas for future research arising from the work presented in this thesis in §9.2.

9.1 Conclusions

In this thesis, we have introduced some relevant preliminary mathematics and reviewed some well-known models and techniques in image processing. We then moved on to consider the application of some image deblurring ideas in Chapter 4 and some enhancements to these models.

In §4.1, we considered the speed and feasibility of implementing the deblurring methods for large images by making use of Fourier transforms and fast solvers. We then considered alternatives to assuming Dirichlet boundary conditions which causes such models to be unsuitable for fast solvers such as conjugate gradient. We overcame this by introducing a new solution strategy using fixed point lagging and preconditioned conjugate gradient which allowed for the functional to be minimised efficiently and for improvements over current models in terms of the quality of the recovered blur function and image to be obtained.

Following this, we presented in §4.2 an accelerated method for reconstructing images in the blind and non-blind cases by making use of variable splitting techniques within the functional to separate the dense blur matrix from the nonlinearity of the regularisation function. This gave us two problems which could each be solved more efficiently than the previous model to give solutions which minimise the functional. Experimental results have demonstrated that, in both the blind and non-blind cases, this new technique is able to obtain good approximations of the true image quickly.

In §4.3, we introduced a new approach to blind deblurring by imposing modified blur function constraints in order to tackle the issue of inaccuracies in the recovered blur function and to alleviate the dependence on heavy regularisation of the blur function, returning emphasis to the task of recovering a blur function which fits the forward problem well. We also introduced a modified hybrid functional which is capable of achieving better results in the case of degradation by Gaussian blur.

We presented the first piece of our main work in Chapter 5 which aimed to obtain improved results in the case of restoring images from known blur degradation by applying constraints on the image intensity values implicitly. We proposed a suitable exponential type transform and applied it to achieve an implicitly constrained solution (positivity at its lower bound and a prescribed intensity value at the upper bound). We presented two solution strategies and a convex variant of the model along with numerical algorithms to solve the resulting non-linear partial differential equations. Test results demonstrated that the proposed methods are able to reconstruct images and point spread functions while keeping the intensity values of the restored data within the appropriate range and is competitive when compared with existing methods.

Building on this, we considered an application of implicit constraints to the more challenging problem of blind deblurring. In Chapter 6, we presented the second piece of our main work which is a core part of this project and addresses the major challenge of how to impose the extra constraints to ensure uniqueness. We proposed two new models and algorithms for implementing blind deconvolution employing the transform based idea, avoiding the commonly used thresholding which leads to unsatisfactory solutions. We also presented numerical results which demonstrate the robustness of the proposed method. We demonstrated that we can ensure positivity and keep the correct range of the image intensities in the case of several blur types, extending the original Chan-Wong model's applicability. This model is particularly effective in reconstructing the kernel without significant defects which can significantly impair the results of previous algorithms.

We aimed to achieve high quality deblurring results in Chapter 7 in a considerably reduced amount of time by making use of available information or assumptions which may be made from visual observation. We achieved this by considering the case of semi-blind deblurring of images which allowed us to recover the full blur function by working with only a small set of parameters. We presented some enhancements to existing work which offer improved results and constructed new formulations which are better able to cope with additive noise in blurred images. Building on the work of Chapter 5, we also incorporated implicitly constrained deblurring into the problem. We demonstrated that this enhanced model offers improved results for Gaussian type blur, particularly in the presence of noise. Following this, we presented formulations to model alternative causes of blur degradation as differentiable functions which may be used in a parametric deblurring framework, allowing such models to be extended beyond the case of Gaussian blur. We demonstrated that this allows us to obtain significantly improved results over alternative parametric methods.

In Chapter 8, we presented the fourth piece of our main work which tackled the well-known and challenging problem of the segmentation of blurred images, which is of great importance. We proposed four variational models, including two models for the simultaneous reconstruction and segmentation of blurred images with spatially invari-

ant blur, without assuming a known blur or blur type. We presented solution methods based on implicitly constrained image reconstruction and convex segmentation. The first methods were aimed at obtaining a good quality segmentation while the others were aimed at improving the speed while retaining the quality. We demonstrated that the new algorithms are effective for segmenting blurred images in the presence of noise and offer improved results over competing models for images corrupted by strong blur.

Some useful future directions arising from this research include the development of methods for automatically selecting optimal regularisation parameters in the case of blind deblurring, further accelerating the solution of the blind deblurring problem, and constructing a convex blind deblurring model. These and other ideas are considered further in the next section.

9.2 Future Work

There are many different directions we can take from the work presented in this thesis. We mention some of them here.

1. We may extend the accelerated model introduced in §4.2 by using Bregman distance or alternate direction methods (ADM) in order to obtain faster convergence. Since the resulting sub-problems would be convex we would have a guarantee of convergence using ADM.

2. In Chapter 5, we spent some time attempting to obtain fast solutions from the resulting highly non-linear model. Although we are able to obtain good results, it would interesting to solve the equation without modifications using fast explicit solution methods which may be developed.

3. A particular goal of Chapter 6 is to remove the harshly implemented optimisation constraints by incorporating them implicitly in the functional. There remain two constraints for unit integral and rotational symmetry which are implemented explicitly at each iteration. While the symmetry is not appropriate for all blur functions, it would be useful to include the unit integral implicitly in the energy functional. The ultimate goal for such models is to present a convex formulation for blind image deconvolution.

4. A possible future direction for the semi-blind deblurring work discussed in Chapter 7 may be to develop solution methods which help to avoid finding local joint minima. More general parametric functions and the incorporation of image statistics to better inform the choice of blur parameter may also be useful.

5. There are many possibilities for the segmentation of blurred images, discussed in Chapter 8. This work can be extended with other aims in mind such as vessel segmentation and infinite perimeter segmentation where the received image is corrupted by blur and noise.

Bibliography

[1] A. Agrawal and R. Raskar. Optimal single image capture for motion deblurring. In *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, pages 2560–2567, 2009.

[2] M. S. C. Almeida and L. B. Almeida. Blind and semi-blind deblurring of natural images. *IEEE T. Image Process.*, 19(1):36–52, 2010.

[3] M. S. C. Almeida and M. A. T. Figueiredo. Parameter estimation for blind and non-blind deblurring using residual whiteness measures. *IEEE T. Image Process.*, 22(7):2751–2763, 2013.

[4] L. Ambrosio and V. M. Tortorelli. On the approximation of free discontinuity problems. *B. Unione Mat. Ital.*, 7(6-B):105–123, 1992.

[5] E. Anarim, H. Ucar, and Y. Istefanopulos. Identification of image and blur parameters in frequency domain using the EM algorithm. *IEEE T. Signal Proces.*, 5(1):159–164, 1996.

[6] P. Arias, V. Caselles, and G. Sapiro. A variational framework for non-local image inpainting. In *Energy Minimization Methods in Computer Vision and Pattern Recognition*, pages 345–358. Springer, 2009.

[7] G. Aubert and P. Kornprobst. *Mathematical Problems in Image Processing: Partial Differential Equations and the Calculus of Variations*, volume 147. Springer, 2006.

[8] N. Badshah and K. Chen. Image selective segmentation under geometric constraints using an active contour approach. *Commun. Comput. Phys.*, 7(4):759–778, 2009.

[9] M. R. Banham and A. K. Katsaggelos. Digital image restoration. *IEEE Signal Proc. Mag.*, 14(2):24–41, 1997.

[10] L. Bar, N. Sochen, and N. Kiryati. Variational pairing of image segmentation and blind restoration. In *Computer Vision-ECCV 2004*, pages 166–177. Springer, 2004.

[11] L. Bar, N. Sochen, and N. Kiryati. Semi-blind image restoration via mumford-shah regularization. *IEEE T. Image Process.*, 15(2):483–493, 2006.

[12] M. Barchiesi, S. H. Kang, T. M. Le, M. Morini, and M. Ponsiglione. A variational model for infinite perimeter segmentations based on lipschitz level set functions: denoising while keeping finely oscillatory boundaries. *Multiscale Model. Simul.*, 8(5):1715–1741, 2010.

[13] J. M. Bardsley and C. R. Vogel. A nonnegatively constrained convex programming method for image reconstruction. *SIAM J. Sci. Comput.*, 25(4):1326–1343, 2004.

[14] B. Bascle, A. Blake, and A. Zisserman. Motion deblurring and super-resolution from an image sequence. In *Computer Vision-ECCV'96*, pages 571–582. Springer, 1996.

[15] F. Benvenuto, R. Zanella, L. Zanni, and M. Bertero. Nonnegative least-squares image deblurring: improved gradient projection approaches. *Inverse Probl.*, 26(2):025004, 2009.

[16] M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester. Image inpainting. In *Proc. 27th Annual Conference on Computer Graphics and Interactive Techniques*, pages 417–424. ACM Press/Addison-Wesley Publishing Co., 2000.

[17] P. J. Bex and W. Makous. Spatial frequency, phase, and the contrast of natural images. *JOSA A*, 19(6):1096–1106, 2002.

[18] J. Biemond, R. L. Lagendijk, and R. M. Mersereau. Iterative methods for image deblurring. *Proc. IEEE*, 78(5):856–883, 1990.

[19] Y. Biraud. A new approach for increasing the resolving power by data processing. *Astron. Astrophys.*, 1:124, 1969.

[20] R. S. Borden. *A Course in Advanced Calculus.* Elsevier Science Publishing, 1983.

[21] K. Bredies, Y. Dong, and M. Hintermüller. Spatially dependent regularization parameter selection in total generalized variation models for image restoration. *Int. J. Comput. Math.*, 90(1):109–123, 2013.

[22] K. Bredies and M. Holler. A tgv regularized wavelet based zooming model. In *Scale Space and Variational Methods in Computer Vision*, volume 7893 of *Lect. Notes Comput. Sc.*, pages 149–160. Springer Berlin Heidelberg, 2013.

[23] K. Bredies, K. Kunisch, and T. Pock. Total generalized variation. *SIAM J. Imaging Sci.*, 3(3):492–526, 2010.

[24] X. Bresson, S. Esedoglu, P. Vandergheynst, J.-P. Thiran, and S. Osher. Fast global minimization of the active contour/snake model. *J. Math. Imaging Vis.*, 28(2):151–167, 2007.

[25] C. Brito-Loeza and K. Chen. Multigrid method for a modified curvature driven diffusion model for image inpainting. *J. Comput. Math.*, 26(6):856–875, 2008.

[26] C. Brito-Loeza and K. Chen. Multigrid algorithm for high order denoising. *SIAM J. Imaging Sci.*, 3(3):363–389, 2010.

[27] E. Brown, T. F. Chan, and X. Bresson. A convex relaxation method for a class of vector valued minimization problems with applications to mumford-shah segmentation. *UCLA CAM Report*, pages 10–43, 2010.

[28] A. Buades, B. Coll, and J.-M. M. Morel. A non-local algorithm for image denoising. In *IEEE Computer Society Conf. Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 60–65, 2005.

[29] A. Buades, B. Coll, and J.-M. M. Morel. A review of image denoising algorithms, with a new one. *Multiscale Model. Simul.*, 4(2):490–530, 2005.

[30] G. J. Burton and I. R. Moorhead. Color and spatial structure in natural scenes. *Appl. Opt.*, 26(1):157–170, 1987.

[31] J.-F. Cai, H. Ji, C. Liu, and Z. Shen. Blind motion deblurring from a single image using sparse approximation. In *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, pages 104–111, 2009.

[32] J.-F. Cai, H. Ji, C. Liu, and Z. Shen. Blind motion deblurring using multiple images. *J. Comput. Phys.*, 228(14):5057–5071, 2009.

[33] J.-F. Cai, H. Ji, C. Liu, and Z. Shen. Framelet based blind motion deblurring from a single image. *IEEE T. Image Process.*, 21(2):562–572, 2012.

[34] D. Calvetti, G. Landi, L. Reichel, and F. Sgallari. Non-negativity and iterative methods for ill-posed problems. *Inverse Probl.*, 20(6):1747–1758, 2004.

[35] D. Calvetti, B. Lewis, L. Reichel, and F. Sgallari. Tikhonov regularization with nonnegativity constraint. *Electron. T. Numer. Ana.*, 18:153–173, 2004.

[36] P. Campisi and K. Egiazarian. *Blind Image Deconvolution*. CRC Press, 2007.

[37] A. S. Carasso. Linear and nonlinear image deblurring: A documented study. *SIAM J. Numer. Anal.*, 36(6):1659–1689, 1999.

[38] A. S. Carasso. Direct blind deconvolution. *SIAM J. Appl. Math.*, 61(6):1980–2007, 2001.

[39] A. Chambolle, V. Caselles, D. Cremers, M. Novaga, and T. Pock. An introduction to total variation for image analysis. *Theoretical foundations and numerical methods for sparse recovery*, 9:263–340, 2010.

[40] A. Chambolle and P.-L. Lions. Image recovery via total variation minimization and related problems. *Numerische Mathematik*, 76(2):167–188, 1997.

[41] A. Chambolle and T. Pock. A first-order primal-dual algorithm for convex problems with applications to imaging. *J. Math. Imaging Vis.*, 40(1):120–145, 2011.

[42] R. H. Chan, T. F. Chan, and C.-K. Wong. Cosine transform based preconditioners for total variation deblurring. *IEEE T. Image Process.*, 8(10):1472–1478, 1999.

[43] R. H. Chan, J. G. Nagy, and R. J. Plemmons. Fft-based preconditioners for toeplitz-block least squares problems. *SIAM J. Numer. Anal.*, 30(6):1740–1768, 1993.

[44] R. H. Chan, M. Tao, and X. Yuan. Constrained total variational deblurring models and fast algorithms based on alternating direction method of multipliers. *SIAM J. Imaging Sci.*, 6(1):680–697, 2013.

[45] R. H. Chan, H. Yang, and T. Zeng. A two-stage image segmentation method for blurry images with poisson or multiplicative gamma noise. *SIAM J. Imaging Sci.*, 7(1):98–127, 2014.

[46] T. Chan, A. Marquina, and P. Mulet. High-order total variation-based image restoration. *SIAM J. Sci. Comput.*, 22(2):503–516, 2000.

[47] T. F. Chan and K. Chen. On a nonlinear multigrid algorithm with primal relaxation for the image total variation minimisation. *Numer. Algorithms*, 41(4):387–411, 2005.

[48] T. F. Chan and K. Chen. An optimization-based multilevel algorithm for total variation image denoising. *Multiscale Model. Simul.*, 5(2):615–645, 2006.

[49] T. F. Chan, S. Esedoglu, and M. Nikilova. Algorithms for finding global minimizers of image segmentation. *SIAM J. Appl. Math.*, 66(5):1632–1648, 2006.

[50] T. F. Chan, G. H. Golub, and P. Mulet. A nonlinear primal-dual method for total variation-based image restoration. *SIAM J. Sci. Comput.*, 20(6):1964–1977, 1999.

[51] T. F. Chan, B. Y. Sandberg, and L. A. Vese. Active contours without edges for vector-valued images. *J. Vis. Commun. Image R.*, 11(2):130–141, 2000.

[52] T. F. Chan and J. J. Shen. Variational image inpainting. *Commun. Pur. Appl. Math.*, 58(5):579–619, 2005.

[53] T. F. Chan and L. A. Vese. Active contours without edges. *IEEE T. Image Process.*, 10(2):266–277, 2001.

[54] T. F. Chan and C.-K. Wong. Total variation blind deconvolution. *IEEE T. Image Process.*, 7(3):370–375, 1998.

[55] T. F. Chan and C.-K. Wong. Convergence of the alternating minimization algorithm for blind deconvolution. *Linear Algebra Appl.*, 316(1):259–285, 2000.

[56] Q. Chang, X.-C. Tai, and L. Xing. A compound algorithm of denoising using second-order and fourth-order partial differential equations. *Numer. Math. Theor. Meth. Appl.*, 2(4):353–376, 2009.

[57] P. Charbonnier, L. Blanc-Féraud, G. Aubert, and M. Barlaud. Deterministic edge-preserving regularization in computed imaging. *IEEE T. Image Process.*, 6(2):298–311, 1997.

[58] J. Chen, L. Yuan, C.-K. Tang, and L. Quan. Robust dual motion deblurring. In *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, pages 1–8. IEEE, 2008.

[59] K. Chen, E. Piccolomini, and F. Zama. An automatic regularization parameter selection algorithm in the total variation model for image deblurring. *Numer. Algorithms*, 67(1):73–92, 2014.

[60] K. Chen. *Matrix Preconditioning Techniques and Applications*. Cambridge University Press, 2005.

[61] K. Chen, S. P. Harding, B. M. Williams, and Y. Zheng. A new study of blind deconvolution with implicit incorporation of non-negativity constraints. *International Journal of Computational Mathematics*, Submitted, 2014.

[62] G. Chung. *A unifying framework for piecewise-constant image segmentation and deblurring*. PhD thesis, University of California, Los Angeles, 2007.

[63] G. Chung and L. A. Vese. Energy minimization based segmentation and denoising using a multilayer level set approach. In *Energy minimization methods in computer vision and pattern recognition (EMMCVPR)*, pages 439–455. Springer, 2005.

[64] A. Cornelio, E. L. Piccolomini, and J. G. Nagy. Constrained numerical optimization methods for blind deconvolution. *Numer. Algorithms*, 65(1):23–42, 2014.

[65] W. R. Crum, O. Camara, and D. L. Hill. Generalized overlap measures for evaluation and validation in medical image analysis. *IEEE T. Med. Imaging*, 25(11):1451–1461, 2006.

[66] N. Dey, L. Blanc-Feraud, C. Zimmer, P. Roux, Z. Kam, J.-C. Olivo-Marin, and J. Zerubia. Richardson–lucy algorithm with total variation regularization for 3d confocal microscope deconvolution. *Microsc. Res. Tech.*, 69(4):260–266, 2006.

[67] L. R. Dice. Measures of the amount of ecologic association between species. *Ecology*, 26(3):297–302, 1945.

[68] M. Donatelli. A multigrid for image deblurring with tikhonov regularization. *Numer. Linear Algebr.*, 12(8):715–729, 2005.

[69] B. Dong, A. Chien, and Z. Shen. Frame based segmentation for medical images. *Communications in Mathematical Sciences*, 9(2):551–559, 2010.

[70] Y. Dong, M. Hintermüller, and M. M. Rincon-Camacho. Automated regularization parameter selection in multi-scale total variation models for image restoration. *J. Math. Imaging Vis.*, 40(1):82–104, 2011.

[71] Y. Duan, Y. Wang, X.-C. Tai, and J. Hahn. A fast augmented lagrangian method for euler's elastica model. In *Scale Space and Variational Methods in Computer Vision*, pages 144–156. Springer, 2012.

[72] P. E. Duda and O. Richard. *Hart, Pattern Classification and Scene Analysis.* John Wiley and Sons, New York, 1973.

[73] M. Elad and A. Feuer. Restoration of a single superresolution image from several blurred, noisy, and undersampled measured images. *IEEE T. Image Process.*, 6(12):1646–1658, 1997.

[74] A. Farcas, L. Elliott, D. B. Ingham, and D. Lesnic. An inverse dual reciprocity method for hydraulic conductivity identification in steady groundwater flow, advances in water resources. *Adv. Water Resour.*, 27(3):223–235, 2004.

[75] D. J. Field. Relations between the statistics of natural images and the response properties of cortical cells. *JOSA A*, 4(12):2379–2394, 1987.

[76] D. J. Field and N. Brady. Visual sensitivity, blur and the sources of variability in the amplitude spectra of natural scenes. *Vision Res.*, 37(23):3367–3383, 1997.

[77] B. Fischer and J. Modersitzki. A unified approach to fast image registration and a new curvature based registration technique. *Linear Algebra Appl.*, 380:107–124, 2004.

[78] D. A. Fish, A. M. Brinicombe, E. R. Pike, and J. G. Walker. Blind deconvolution by means of the richardson-lucy algorithm. *J. Opt. Soc. Am. A*, 12(1):58–65, 1995.

[79] A. Foi, M. Trimeche, V. Katkovnik, and K. Egiazarian. Practical poissonian-gaussian noise model and fitting for single-image raw-data. *IEEE T. Image Process.*, 17(10):1737–1754, 2008.

[80] S. Fuhrmann and M. Goesele. Fusion of depth maps with multiple scales. In *ACM Transactions on Graphics (TOG)*, volume 30(6), page 148. ACM, 2011.

[81] I. M. Gelfand and S. V. Fomin. *Calculus of Variations*. Prentice-Hall, Inc., New Jersey, USA, 1963.

[82] B. Ghanbari, L. Rada, and K. Chen. A restarted iterative homotopy analysis method for two nonlinear models from image processing. *Int. J. Comput. Math.*, 91(3):661–687, 2014.

[83] M. Giaquinta and S. Hildebrandt. *Calculus of Variations I, The Lagrangian Formalism*. Springer-Verlag, 1996.

[84] M. Giaquinta and S. Hildebrandt. *Calculus of Variations II, The Hamiltonian Formalism*. Springer-Verlag, 1996.

[85] T. Goldstein, X. Bresson, and S. Osher. Geometric applications of the split bregman method: Segmentation and surface reconstruction. *J. Sci. Comput.*, 45(1-3):272–293, 2010.

[86] T. Goldstein and S. Osher. The split bregman method for l1-regularized problems. *SIAM J. Imaging Sci.*, 2(2):323–343, 2009.

[87] G. H. Golub and C. F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, Baltimore, 1996.

[88] G. H. Golub and Q. Ye. Inexact preconditioned conjugate gradient method with inner-outer iteration. *SIAM J. Sci. Comput.*, 21(4):1305–1320, 1997.

[89] C. Gout, C. Le Guyader, and L. A. Vese. Segmentation under geometrical conditions using geodesic active contours and interpolation using level set methods. *Numer. Algorithms*, 39(1-3):155–173, 2005.

[90] J. Hadamard. Sur les problèmes aux dérivées partielles et leur signification physique. *Princeton University Bulletin*, 13(49-52):28, 1902.

[91] M. Hanke. Limitations of the l-curve method in ill-posed problems. *BIT*, 36(2):287–301, 1996.

[92] B. C. Hansen and R. F. Hess. Discrimination of amplitude spectrum slope in the fovea and parafovea and the local amplitude distributions of natural scene imagery. *J. Vis.*, 6(7):696–711, 2006.

[93] C. Hansen. Analysis of discrete ill-posed problems by means of the l-curve. *SIAM Rev.*, 34(4):561–580, 1992.

[94] C. Hansen, J. G. Nagy, and D. P. O'Leary. *Deblurring Images: Matrices, Spectra, and Filtering*. SIAM Publications, 2006.

[95] C. Hansen and D. P. OLeary. The use of the l-curve in the regularization of discrete ill-posed problems. *SIAM J. Sci. Comput.*, 14(6):1487–1503, 1993.

[96] S. W. Hasinoff, F. Durand, and W. T. Freeman. Noise-optimal capture for high dynamic range photography. In *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, pages 553–560, 2010.

[97] S. W. Hasinoff, K. N. Kutulakos, F. Durand, and W. T. Freeman. Time-constrained photography. In *IEEE 12th Int. Conf. Computer Vision (ICCV)*, pages 333–340, 2009.

[98] M. R. Hestenes and E. Stiefel. Methods of conjugate gradients for solving linear systems. *J. Res. Nat. Bur. Stand.*, 49(6):409–436, December 1952.

[99] N. J. Higham. *Accuracy and Stability of Numerical Algorithms*. SIAM, 2002.

[100] J.-B. Hiriart-Urruty and C. Lemaréchal. *Convex Analysis and Minimization Algorithms: Part 1: Fundamentals*, volume 305 of *A Series of Comprehensive Studies in Mathematics*. Springer, 1993.

[101] S. Hoffmann, M. Mainberger, J. Weickert, and M. Puhl. Compression of depth maps with segment-based homogeneous diffusion. In *Scale Space and Variational Methods in Computer Vision*, pages 319–330. Springer, 2013.

[102] T. S. Huang, W. Schreiber, and O. J. Tretiak. Image processing. *Proc. IEEE*, 59(11):1586–1609, 1971.

[103] Y. Huang and M. K. Ng. Lipschitz and total-variational regularization for blind deconvolution. *Commun. Comput. Phys.*, 4:195–206, 2008.

[104] Y. Huang, M. K. Ng, and Y.-W. Wen. A fast total variation minimization method for image restoration. *Multiscale Model. Simul.*, 7(2):774–795, 2008.

[105] P. J. Huber. *Robust statistics*. Springer, 2011.

[106] M. Ibrahim and K. Chen. A composition model and its algorithm combining parametric transformation and non-parametric deformation for effective image registration. Submitted, 2014.

[107] M. Irani and S. Peleg. Improving resolution by image registration. *CVGIP-Graph Model. Im.*, 53(3):231–239, 1991.

[108] P. Jaccard. Nouvelles recherches sur la distribution florale. *Bull. Soc. Vaudoise Sci. Nat.*, 44:223–270, 1908.

[109] M. Jiang, G. Wang, M. W. Skinner, J. T. Rubinstein, and M. W. Vannier. Blind deblurring of spiral ct images. *IEEE T. Med. Imaging*, 22(7):837–845, 2003.

[110] H. J. Johnson and G. E. Christensen. Consistent landmark and intensity-based image registration. *IEEE T. Med. Imaging*, 21(5):450–461, 2002.

[111] M. Jung, G. Chung, G. Sundaramoorthi, L. A. Vese, and A. L. Yuille. Sobolev gradients and joint variational image segmentation, denoising, and deblurring. In *IS&T/SPIE Electronic Imaging*, pages 72460I–72460I. International Society for Optics and Photonics, 2009.

[112] R. Kaftory, N. Sochen, and Y. Y. Zeevi. Variational blind deconvolution of multi-channel images. *Int. J. Imag. Syst. Tech.*, 15(1):56–63, 2005.

[113] J. Kim, A. Tsai, M. Cetin, and A. S. Willsky. A curve evolution-based variational approach to simultaneous image restoration and segmentation. In *IEEE Int. Conf. Image Processing (ICIP)*, volume 1, pages I–109, 2002.

[114] R. Kimmel, R. Malladi, and N. Sochen. Images as embedded maps and minimial surfaces: Movies, colour, texture and volumetric medical images. *Int. J. Comput. Vision*, 39(2):111–129, 2000.

[115] M. Klodt and D. Cremers. A convex framework for image segmentation with moment constraints. In *IEEE Int. Conf. Computer Vision (ICCV)*, pages 2236–2243, 2011.

[116] F. Knoll, K. Bredies, T. Pock, and R. Stollberger. Second order total generalized variation (tgv) for mri. *Magn. Reson. Med.*, 65(2):480–491, 2011.

[117] D. Kundur and D. Hatzinakos. Blind image deconvolution. *IEEE Signal Proc. Mag.*, 13(3):43–64, 1996.

[118] D. Kundur and D. Hatzinakos. Blind image deconvolution revisited. *IEEE Signal Proc. Mag.*, 13(6):61–63, 1996.

[119] R. L. Lagendijk, I. Biemond, and D. E. Boekee. Regularized iterative image restoration with ringing reduction. *IEEE T. Acoust. Speech*, 36(12):1874–1888, 1988.

[120] R. L. Lagendijk, J. Biemond, and D. E. Boekee. Identification and restoration of noisy blurred images using the expectation-maximization algorithm. *IEEE T. Acoust. Speech*, 38(7):1180–1191, 1990.

[121] R. L. Lagendijk, A. M. Tekalp, and J. Biemond. Maximum likelihood image and blur identification: A unifying approach. *Opt. Eng.*, 29(5):422–435, 1990.

[122] J. Lellmann and C. Schnörr. Continuous multiclass labeling approaches and algorithms. *SIAM J. Imaging Sci.*, 4(4):1049–1096, 2011.

[123] A. Levin. Blind motion deblurring using image statistics. In *Advances in Neural Information Processing Systems (NIPS)*, 2007.

[124] A. Levin, Y. Weiss, F. Durand, and W. T. Freeman. Understanding and evaluating blind deconvolution algorithms. In *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, pages 1964–1971, 2009.

[125] A. Levin, Y. Weiss, F. Durand, and W. T. Freeman. Efficient marginal likelihood optimization in blind deconvolution. In *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, pages 2657–2664, 2011.

[126] A. Levin, Y. Weiss, F. Durand, and W. T. Freeman. Understanding blind deconvolution algorithms. *IEEE T. Pattern Anal.*, 33(12):2354–2367, 2011.

[127] C. Li, C.-Y. Kao, J. C. Gore, and Z. Ding. Minimization of region-scalable fitting energy for image segmentation. *IEEE T. Image Process.*, 17(10):1940–1949, 2008.

[128] S. Li and B. Yang. Multifocus image fusion using region segmentation and spatial frequency. *Image Vision Comput.*, 26(7):971–979, 2008.

[129] C. Liu, R. Szeliski, S. B. Kang, C. L. Zitnick, and W. T. Freeman. Automatic estimation and removal of noise from a single image. *IEEE T. Pattern Anal.*, 30(2):299–314, 2008.

[130] H. Liu, Y. Zhang, and S. Ji. Study on the methods of super-resolution image reconstruction, 2008.

[131] R. Liu, Z. Li, and J. Jia. Image partial blur detection and classification. In *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, pages 1–8, 2008.

[132] T. Lu, P. Neittaanmäki, and X.-C. Tai. A parallel splitting-up method for partial differential equations and its applications to navier-stokes equations. *Math. Modell. Numer. Anal.*, 26(6):673–708, 1992.

[133] M. Lysaker, A. Lundervold, and X.-C. Tai. Noise removal using fourth-order partial differential equation with applications to medical magnetic resonance images in space and time. *IEEE T. Image Process.*, 12(12):1579–1590, 2003.

[134] M. Lysaker and X.-C. Tai. Iterative image restoration combining total variation minimization and a second-order functional. *Int. J. Comput. Vision*, 66(1):5–18, 2006.

[135] S. Makni, P. Ciuciu, J. Idier, and J.-B. Poline. Semi-blind deconvolution of neural impulse response in fmri using a gibbs sampling method. In *IEEE Int. Conf. Acoustics, Speech, and Signal Processing (ICASSP)*, volume 5, pages V–601, 2004.

[136] L. Marin, L. Elliott, P. J. Heggs, D. B. Ingham, D. Lesnic, and X. Wen. Boundary element solution for the cauchy problem associated with the helmholtz equation by the tikhonov regularisation method. In *Inverse problems in engineering mechanics IV: International Symposium on Inverse Problems in Engneering Mechanics (ISIP)*, pages 485–494, Nagano, Japan, 2003. Elsevier Science Ltd.

[137] O. Michailovich. Spatially regularized compressed sensing for high angular resolution diffusion imaging. *IEEE T. Med. Imaging*, 30(5):1100–1115, 2011.

[138] N. Miura. Blind deconvolution under band limitation. *Optics letters*, 28(23):2312–2314, 2003.

[139] J. Modersitzki. *Numerical methods for image registration, (Numerical mathematics & scientific computation)*. Oxford University Press, 2004.

[140] J. Modersitzki. *FAIR: Flexible algorithms for image registration*. SIAM Publications, 2009.

[141] R. Molina, J. Mateos, and A. Katsaggelos. Blind deconvolution using a variational approach to parameter, image, and blur estimation. *IEEE T. Image Process.*, 15(12):3715–3727, 2006.

[142] J. H. Money and S. H. Kang. Total variation minimizing blind deconvolution with shock filter reference. *Image Vision Comput.*, 26(2):302–314, 2008.

[143] D. Mumford and J. Shah. Optimal approximation by piecewise smooth functions and associated variational problems. *Commun. Pur. Appl. Math.*, 42(5):577–685, 1989.

[144] S. K. Nayar and M. Ben-Ezra. Motion-based motion deblurring. *IEEE T. Pattern Anal.*, 26(6):689–698, 2004.

[145] A. Nemirovski. Lectures on modern convex optimization. In *SIAM*. Citeseer, 2001.

[146] M. K. Ng and N. K. Bose. Mathematical analysis of super-resolution methodology. *IEEE Signal Proc. Mag.*, 20(3):62–74, 2003.

[147] M. K. Ng, R. H. Chan, and W.-C. Tang. A fast algorithm for deblurring models with neumann boundary conditions. *SIAM J. Sci. Comput.*, 21(3):851–866, 1999.

[148] T. N. A. Nguyen, J. Cai, J. Zhang, and J. Zheng. Robust interactive image segmentation using convex active contours. *IEEE T. Image Process.*, 21(8):3734–3743, 2012.

[149] J. P. Oliveira, J. M. Bioucas-Dias, and M. A. Figueiredo. Adaptive total variation image deblurring: A majorization – minimization approach. *Signal Process.*, 89(9):1683–1693, 2009.

[150] J. M. Ortega and W. C. Rheinboldt. *Iterative Solution of Nonlinear Equations in Several Variables*. Academic Press, New York, USA, 1970.

[151] S. Osher. and J. A. Sethian. Fronts propagating with curvature-dependent speed: Algorithms based on hamiltonjacobi formulations. *J. Comput. Phys.*, 79(1):12–49, 1988.

[152] S. Osher, M. Burger, D. Goldfarb, J. Xu, and W. Yin. An iterative regularization method for total variation-based image restoration. *Multiscale Model. Simul.*, 4(2):460–489, 2005.

[153] K. Papafitsoros and C.-B. Schönlieb. A combined first and second order variational approach for image reconstruction. *J. Math. Imaging Vis.*, 48(2):308–338, 2014.

[154] S. U. Park, N. Dobigeon, and A. O. Hero. Semi-blind sparse image reconstruction with application to mrfm. *IEEE T. Image Process.*, 21(9):3838–3849, 2012.

[155] V. Pascazio. Statistical regularization in linearized microwave imaging through mrf-based map estimation: hyperparameter estimation and image computation. *IEEE T. Image Process.*, 12(5):572–582, 2003.

[156] D. Perrone and P. Favaro. Total variation blind deconvolution: The devil is in the details. In *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, pages 2909–2916. IEEE, 2014.

[157] N. Persch, A. Elhayek, M. Welk, A. Bruhn, S. Grewenig, K. Böse, A. Kraegeloh, and J. Weickert. Enhancing 3-d cell structures in confocal and sted microscopy: a joint model for interpolation, deblurring and anisotropic smoothing. *Meas. Sci. Technol.*, 24(12):125703, 2013.

[158] P. Peter. Three-dimensional data compression with anisotropic diffusion. In *Pattern Recognition*, pages 231–236. Springer, 2013.

[159] G. Piella. New quality measures for image fusion. In *Proc. 7th Int. Conf. Information Fusion*, pages 542–546, 2004.

[160] G. Pillonetto and C. Cobelli. Identifiability of the stochastic semi-blind deconvolution problem for a class of time-invariant linear systems. *Automatica*, 43(4):647–654, 2007.

[161] T. Pock, D. Cremers, H. Bischof, and A. Chambolle. A convex relaxation approach for computing minimal partitions. In *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, pages 810–817, 2009.

[162] L. Rada and K. Chen. A new variational model with dual level set functions for selective segmentation. *Commun. Comput. Phys.*, 12(1):261–283, 2012.

[163] L. Rada and K. Chen. On a variational model for selective image segmentation of features with infinite perimeter. *Journal of Mathematical Research with Applications*, 33(3):253–272, 2013.

[164] A. Rap, L. Elliott, D. B. Ingham, D. Lesnic, and X. Wen. The inverse source problem for the variable coefficients convection-diffusion equation. *Inverse Probl. Sci. En.*, 15(5):413–440, 2007.

[165] A. Rav-Acha and S. Peleg. Two motion-blurred images are better than one. *Pattern Recogn. Lett.*, 26(3):311–317, 2005.

[166] R. Reddy, M. Chandra, and R. Rao. Segmentation of blurred images using improved chan-vese snake model. In *Int. Conf. Signal Processing, Communication, Computing and Networking Technologies (ICSCCN)*, pages 502–505, 2011.

[167] W. H. Richardson. Bayesian-based iterative method of image restoration. *JOSA*, 62(1):55–59, 1972.

[168] S. Roth and M. J. Black. Fields of experts: A framework for learning image priors. In *IEEE Computer Society Conf. Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 860–867, 2005.

[169] L. Rudin, S. Osher, and E. Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D*, 60(1):259–268, 1992.

[170] W. Rudin. *Principles of Mathematical Analysis (International Series in Pure & Applied Mathematics)*. McGraw-Hill Publishing Co., 1976.

[171] D. Rueckert, L. I. Sonoda, C. Hayes, D. L. G. Hill, M. O. Leach, and D. J. Hawkes. Nonrigid regisration using free-form deformations: applications to breast mr images. *IEEE T. Med. Imaging*, 18(8):712–721, 1999.

[172] P. Sarri, G. Thomas, E. Sekko, and P. Neveux. Myopic deconvolution combining kalman filter and tracking control. In *IEEE Int. Conf. Acoustics, Speech and Signal Processing*, volume 3, pages 1833–1836, 1998.

[173] J. Savage and K. Chen. On multigrids for solving a class of improved total variation based staircasing reduction models. In *Image Processing Based on Partial Differential Equations*, pages 69–94. Springer, 2007.

[174] A. Sawatzky, C. Brune, J. Müller, and M. Burger. Total variation processing of images with poisson statistics. In *Computer Analysis of Images and Patterns*, pages 533–540. Springer, 2009.

[175] C. Schmaltz, P. Peter, M. Mainberger, F. Ebel, J. Weickert, and A. Bruhn. Understanding, optimising, and extending data compression with anisotropic diffusion. *Int. J. Comput. Vision*, 108(3):1–19, 2014.

[176] S. Setzer, G. Steidl, and T. Teuber. Deblurring poissonian images by split bregman techniques. *J. Vis. Commun. Image R.*, 21(3):193–199, 2010.

[177] M. I. Sezan and A. M. Tekalp. Survey of recent developments in digital image restoration. *Opt. Eng.*, 29(5):393–404, 1990.

[178] M. Sezan and H. Trussell. Prototype image constraints for set-theoretic image restoration. *IEEE T. Signal Proces.*, 39(10):2275–2285, 1991.

[179] Q. Shan, J. Jia, and A. Agarwala. High-quality motion deblurring from a single image. *ACM Trans. Graph.*, 27(3):73:1–73:10, 2008.

[180] Y. Shi, Q. Chang, and J. Xu. Convergence of fixed point iteration for deblurring and denoising problem. *Appl. Math. Comput.*, 189(2):1178–1185, 2007.

[181] D. L. Snyder, J. O'Sullivan, B. Whiting, and R. Murphy. Deblurring subject to nonnegativity constraints when known functions are present with application to object-constrained computerized tomography. *IEEE T. Med. Imaging*, 20(10):1009–1017, 2001.

[182] D. L. Snyder, T. Schulz, and A. O'Sullivan. Deblurring subject to nonnegativity constraints. *IEEE T. Signal Proces.*, 40(5):1143–1150, 1992.

[183] A. Spira, N. Sochen, and R. Kimmel. *Geometric filters, diffusion flows, and kernels in image processing, In Eduardo Bayro-Corrochano (Editor), Handbook of computational geometry for pattern recognition, computer vision, neurocomputing and robotics*. Springer-Verlag, Berlin, 2005.

[184] F. Sroubek and J. Flusser. Multichannel blind iterative image restoration. *IEEE T. Image Process.*, 12(9):1094–1106, 2003.

[185] F. Sroubek and P. Milanfar. Robust multichannel blind deconvolution via fast alternating minimization. *IEEE T. Image Process.*, 21(4):1687–1700, 2012.

[186] T. G. Stockham Jr, T. M. Cannon, and R. B. Ingebretsen. Blind deconvolution through digital signal processing. *Proc. IEEE*, 63(4):678–692, 1975.

[187] L. Y. Su and R. H. Liu. Blind image restoration with modified CMA. *Int. J. Image Grap.*, 11(03):403–414, 2011.

[188] L. Sun and K. Chen. A new iterative algorithm for mean curvature-based variational image denoising. *BIT*, pages 1–31, 2013.

[189] P. Thevanaz, U. E. Ruttimann, and M. Unser. A pyramid approach to subpixel registration based on intensity. *IEEE T. Image Process.*, 7(1):27–41, 1998.

[190] A. Tikhonov and V. Arsenin. *Solutions of Ill-Posed Problems*. Winston and Sons, New York, 1977.

[191] D. Tolhurst, Y. Tadmor, and T. Chao. Amplitude spectra of natural images. *Ophthalmic Physiol. Opt.*, 12(2):229–232, 1992.

[192] T. Treibitz and Y. Schechner. Polarization: Beneficial for visibility enhancement? In *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, pages 525–532, 2009.

[193] C. R. Vogel. *Computational Methods for Inverse Problems*. SIAM, 2002.

[194] C. R. Vogel and M. E. Oman. Fast, robust total variation-based reconstruction of noisy, blurred images. *IEEE T. Image Process.*, 7(6):813–824, 1998.

[195] C. R. Vogel and M. E. Oman. Fast total variation-based image reconstruction. In *Proc. 1995 ASME Design Engineering Conferences*, volume 3, part C, pages 1009–1015, 1995.

[196] C. R. Vogel and M. E. Oman. Iterative methods for total variation denoising. *SIAM J. Sci. Comput.*, 17(1):227–238, 1996.

[197] F. Wang. *Alternating Direction Methods for Image Recovery*. PhD thesis, Hong Kong Baptist University, 2012.

[198] W. Wang and M. K. Ng. On algorithms for automatic deblurring from a single image. *J. Comput. Math.*, 30(1):80–100, 2012.

[199] Z. Wang and A. C. Bovik. A universal image quality index. *IEEE Signal Proc. Let.*, 9(3):81–84, 2002.

[200] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE T. Image Process.*, 13(4):600–612, 2004.

[201] Z. Wang and Q. Li. Information content weighting for perceptual image quality assessment. *IEEE T. Image Process.*, 20(5):1185–1198, 2011.

[202] G. W. Wei. Generalized perona-malik equation for image restoration. *IEEE Signal Proc. Let.*, 6(7):165–167, 1999.

[203] J. Weickert, B. T. H. Romeny, and M. A. Viergever. Efficient and reliable schemes for nonlinear diffusion filtering. *IEEE T. Image Process.*, 7(3):398–410, 1998.

[204] M. Welk. Robust variational approaches to positivity-constrained image deconvolution. Technical report, Universität des Saarlandes, 2010.

[205] Y.-W. Wen and R. H. Chan. Parameter selection for total variation based image restoration using discrepancy principle. *IEEE T. Image Process.*, 21(4):1770–1781, 2012.

[206] O. Whyte, J. Sivic, A. Zisserman, and J. Ponce. Non-uniform deblurring for shaken images. *Int. J. Comput. Vision*, 98(2):168–186, 2012.

[207] B. M. Williams, K. Chen, and S. P. Harding. A new constrained total variational deblurring model and its fast algorithm. *Numer. Algorithms*, Accepted, 2014.

[208] B. M. Williams, K. Chen, Y. Zheng, and S. P. Harding. Advanced blur removal methods with applications to retinal imaging for ophthalmology. In X. Xie, editor, *Medical Image Understanding and Analysis 2012*, Medical Image Understanding and Analysis, pages 286–291, July 2012.

[209] B. M. Williams, K. Chen, Y. Zheng, and S. P. Harding. A blind deconvolution model for images corrupted by gaussian blur. In *Essex-Greenwich-Hertfordshire Workshop on Applied and Numerical Mathematics - Multiscale Problems*, 2012.

[210] B. M. Williams, K. Chen, Y. Zheng, and S. P. Harding. Mathematical deblurring of images for non-blind and blind restoration. In *25th Biennial Numerical Analysis Conference*, 2013.

[211] B. M. Williams, K. Chen, Y. Zheng, and S. P. Harding. An enhanced model for semi-blind deblurring with parametric kernel identification and constrained image reconstruction. *Submitted*, 2014.

[212] B. M. Williams, J. A. Spencer, K. Chen, Y. Zheng, and S. P. Harding. An effective variational model for simultaneous reconstruction and segmentation of blurred images. *Computer Vision Image and Understanding Special Issue on Discrete and Continuous Geometry in Computer Vision*, Submitted, 2014.

[213] H. Xu, Q. Sun, N. Luo, G. Cao, and D. Xia. Iterative nonlocal total variation regularization method for image restoration. *PLoS ONE*, 8(6):e65865, 2013.

[214] Y. Yitzhaky and N. S. Kopeika. Identification of blur parameters from motion blurred images. *Graph. Model. Im. Proc.*, 59(5):310–320, 1997.

[215] Y.-L. You and M. Kaveh. Fourth-order partial differential equations for noise removal. *IEEE T. Image Process.*, 9(10):1723–1730, 2000.

[216] Y.-L. You and M. M. Kaveh. A regularization approach to joint blur identification and image restoration. *IEEE T. Image Process.*, 5(3):416–428, 1996.

[217] W. L. Zeng and X. B. Lu. A robust variational approach to super-resolution with nonlocal tv regularisation term. *Imaging Sci. J.*, 61(2):268–278, 2013.

[218] J. Zhang, K. Chen, and B. Yu. An iterative lagrange multiplier method for constrained total variation-based image denoising. *SIAM J. Numer. Anal.*, 50(3):983–1003, 2012.

[219] X. Zhang, M. Burger, X. Bresson, and S. Osher. Bregmanized nonlocal regularization for deconvolution and sparse reconstruction. *SIAM J. Imaging Sci.*, 3(3):253–276, 2010.